SORD

M243EX Series

MDOS/XMS
(OPERATING SYSTEM)

FIRST EDITION 1983.9

SORD COMPUTER CORPORATION

Scanned with CamScanner

PREFACE

This manual describes functions and operations of MDOS/XMS, one of general-purpose disk operating systems for the SORD M243EX series computer. To ensure that a maximum number of readers can use this manual profitably, these functions and operations are explained extensively.

It consists of the following eight chapters:

- CHAPTER 1 GENERAL

  Describes MDOS/XMS features and outlines its functions.

- CHAPTER 2 FILE MANAGEMENT

  Describes file management functions to be understood before using MDOS/XMS.

- CHAPTER 3 CONSOLE CONTROL

  Describes console control functions for the keyboard and display unit to be understood before using MDOS/XMS.

- CHAPTER 4 MULTIJOB PROCESSING

  Describes how multijob processing of foreground and background jobs is performed under control of MDOS/XMS.

- CHAPTER 5 MDOS/XMS BOOTSTRAP PROCEDURE

  Describes the procedure for bootstrapping MDOS/XMS into the M243EX mark series computer system.

- CHAPTER 6 COMMAND LINE INTERPRETER (CLI)

  Describes how to use the Command Line Interpreter (CLI) which is the interface between the user and MDOS/XMS.

- CHAPTER 7 CLI COMMANDS

  Describes functions, entry formats, and examples of CLI commands used to communicate with MDOS/XMS.

- CHAPTER 8 PROCESSING PROGRAMS

  Describes various processing program provided together with MDOS/XMS. For details on these processing programs, the user should refer to the appropriate manuals.

This manual does not describe M243EX mark series computer hardware functions. For details on these hardware functions, see the following manual:

- M243EX Mark Series User's Guide

CONTENTS

		Page
CHAPTER 1	GENERAL	1
1.1	MDOS/XMS Features	1
1.2	MDOS/XMS Configuration	2
1.3	Command Line Interpreter (CLI)	7
1.4	System Call	9
1.5	Files	11
1.6	Concept of Multijob Processing	13
1.7	Disk File Types	13
CHAPTER 2	FILE MANAGEMENT	17
2.1	Memory Files	17
2.2	Disk Files	21
2.3	Directories	25
2.4	File Access Methods	29
CHAPTER 3	CONSOLE CONTROL	32
3.1	Keyboard	32
3.2	Display Unit	39

		Page
CHAPTER 4	MULTIJOB PROCESSING	43
4.1	Purposes of Multijob Processing	43
4.2	Multijob Control	45
4.3	Job Types and Activating Jobs	47
CHAPTER 5	MDOS/XMS BOOTSTRAP PROCEDURE	52
CHAPTER 6	COMMAND LINE INTERPRETER (CLI)	57
6.1	Overview	57
6.2	Entry of More Than One CLI Command	58
6.3	Indirect Specification of Command Lines..	59
6.4	Specifying a Disk File	60
CHAPTER 7	CLI COMMANDS	65
7.1	BDIR	65
7.2	CDIR	66
7.3	CHATR	68
7.4	COPY	70
7.5	CREATE	78
7.6	CREV	80
7.7	DATE	82
7.8	DDIR	84
7.9	DELETE	85
7.10	DIRT	89
7.11	DISK	90
7.12	DUMP	91

Page

7.13 ERMSG 93

7.14 FILE 94

7.15 FUNC 96

7.16 LIST 101

7.17 MINIT 104

7.18 OSCHG 107

7.19 PASS 109

7.20 POFF 110

7.21 RENAME 111

7.22 REZERO 112

7.23 SGL 113

7.24 SIO 115

7.25 SLICE 118

7.26 TIME 119

7.27 VOL 121

7.28 VCOPY 125

7.29 XFER 129

		Page
CHAPTER 8	PROCESSING PROGRAMS	137
8.1	System Programs	137
8.1.1	BASIC	138
8.1.2	RASM	139
8.1.3	RLDR	142
8.1.4	DBG	145
8.1.5	LFE	150
8.1.6	EDIT	152
8.1.7	FORTRAN	161
8.1.8	COBOL	164
8.1.9	PIPS	168
8.1.10	SGL	168
8.2	Utility Programs	169
8.2.1	INIT	169
8.2.2	SIFT	173
8.2.3	SST-1	176
8.2.4	TAPPER	180
8.2.5	MSORT	184
8.2.6	SYSGEN	189
8.2.7	MTBK and MTDL	209
8.2.8	MDUMP	210
8.2.9	MLOAD	215
8.2.10	ADUMP	218
8.2.11	ALOAD	222
8.2.12	ACOMP	224
APPENDIX A	MAIN FRAME DIP SWITCHES	226

CHAPTER 1 GENERAL

1.1 MDOS/XMS Features

The MDOS/XMS disk operating system was developed to realize multijob processing and multiple language processing on a SORD M243EX mark series computer, that is, enable two or more jobs written in different programming languages to be processed concurrently on one SORD M243EX mark series computer. Main MDOS/XMS features are as follows:

- The standard M243EX mark series computer hardware is provided with three 64K bytes memory banks. One of the three memory banks is used as the OS resident memory area and the others are used for jobs prepared by the user. The number of user jobs which can be processed concurrently can be increased by adding optional memory.

- MDOS/XMS allows the user to write jobs in different programming languages because it allocates one 64K byte memory bank to each job.

- To enable higher-speed data transfer between memory and external storage devices, MDOS/XMS processes data in units of 512 bytes in input channels and I/O buffers.

- MDOS/XMS provides the memory file function so as to expedite communication between larger programs written in BASIC (CHAIN function). The memory file function enables frequently used data to be loaded to the program area at a higher speed because it allows a file on a memory bank to be stored in the same manner as on an external storage device.

. To support various I/O devices, MDOS/XMS controls four serial I/O ports by assigning a unique I/O file name to each port. It also assigns an I/O file name to each console (comprising of a keyboard and a display unit) of an M243EX mark series computer system for control.

1.2 MDOS/XMS Configuration

MDOS/XMS was designed to make an M243EX mark series personal computer system operate most efficiently. For this purpose, MDOS/XMS is separated into modules. MDOS/XMS consists of the following main units:

MDOS/XMS

- Memory control
- Job control
- File management
- Time control
- Console control
- Disk control
- System call control

Memory control

A 64K byte memory space is called a memory bank and MDOS/XMS controls the M243EX mark series computer memory in units of a memory bank. As shown in Figure 1.1, MDOS/XMS is resident on Memory bank 0 and the leading portion of the bank is used for the interrupt vector table and message exchange between jobs.

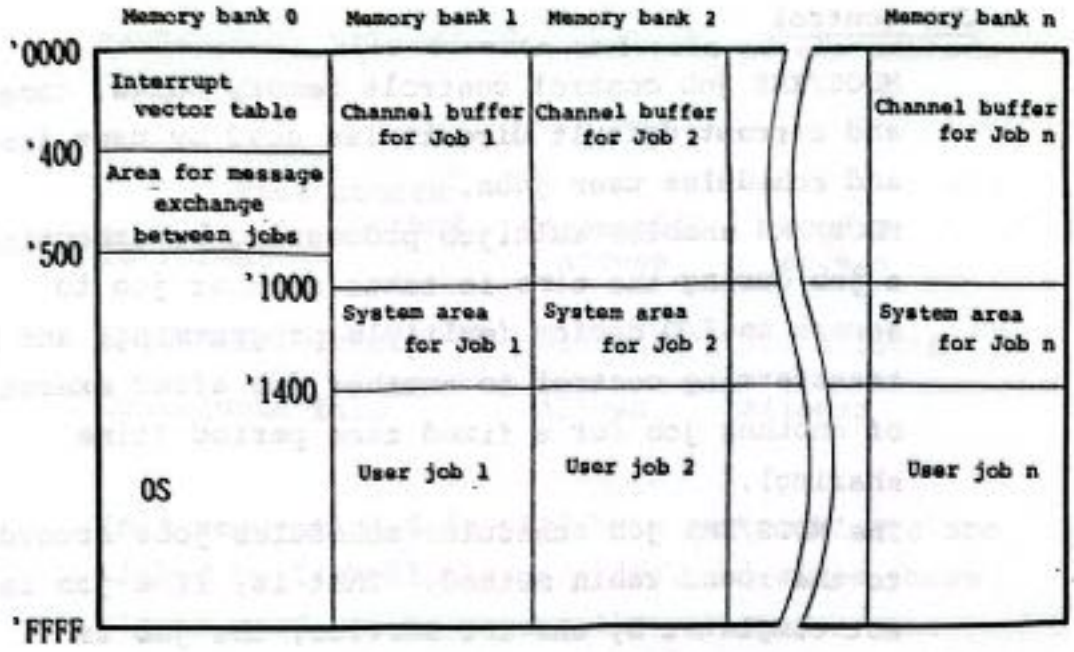


Fig. MDOS/XMS memory spaces

MDOS/XMS allocates one memory bank to each user job and checks whether which memory bank is being used or not. When the user is activating a job, memory control searches for a unused memory bank and allocates it to the job. Therefore, the user can activate a job without checking the location and availability of each memory bank.

Job control

MDOS/XMS job control controls memory banks, consoles, and current default directories used by user jobs, and schedules user jobs.

MDOS/XMS enables multijob processing by executing a job during the time it takes another job to access an I/O device (multiple programming) and by transferring control to another job after execution of another job for a fixed time period (time sharing).

The MDOS/XMS job scheduler schedules jobs according to the round robin method. That is, if a job is not completed by one CPU service, the job is entered in the last of the job execution queue and after all other executable jobs receive a CPU service, it is serviced again.

File management

MDOS/XMS controls file spaces and enables the user to access files. Furthermore, it allows to the user to use free memory banks as file spaces as well as disks and I/O devices.

MDOS/XMS supports two disk file formats: sequential and contiguous, and two file access methods: sequential and random.

Table File formats and file access methods

File format	File access method: Sequential access	Random access
Sequential access	Allowed	Not allowed
Contiguous file	Allowd	Allowed

In a sequential file, 512 byte sector blocks are linked by link blocks on a disk and since these blocks need not be physically continuous, the file can be extended easily. A contiguous file consists of physically continuous blocks, so it cannot be extended.

In the sequential access method, MDOS/XMS indicates on which byte (from the first byte) the file access starts, using the file pointer.

In addition, MDOS/XMS allows the user to move the file pointer. By this function, the user can rewrite data in a sequential file freely, moving the file pointer.

The random access method is used to access record in a file and can be used only for contiguous files.

MDOS/XMS protects files according to their attributes. To protect important data and programs, the user should use passwords for their files and directories.

File management provides file sharing and monopolization modes.

Time control

The M243EX mark series computer is provided with a Real Time Clock (RTC) for controlling the current time and a Clock Timer Counter (CTC) for controlling internal operations. MDOS/XMS displays the current time and indicates file creation times on the console screen by using the RTC, and controls the time slice for jobs and monitors I/O devices for timeout errors by using the CTC.

Console control

In addition to control of the current console status, MDOS/XMS decodes key input from the keyboard and controls special function keys. It also controls character sets and attributes to be displayed on the screen.

Disk control

MDOS/XMS controls disk spaces in units of a block by handling 512 bytes (sector) as one block and assigning a sequential number (logical block number) to each block from zero. Therefore, even if different types of disk are used, MDOS/XMS can control their spaces in the same manner as written above only if an appropriate disk handler is added.
MDOS/XMS provides each disk with a block called a volume block and allows the user to write a disk user name, a creation date, a purpose, an application, etc., on it.

System call control

MDOS/XMS provides the system call function for linking it to a user program. This function uses the CPU RESTART instruction.

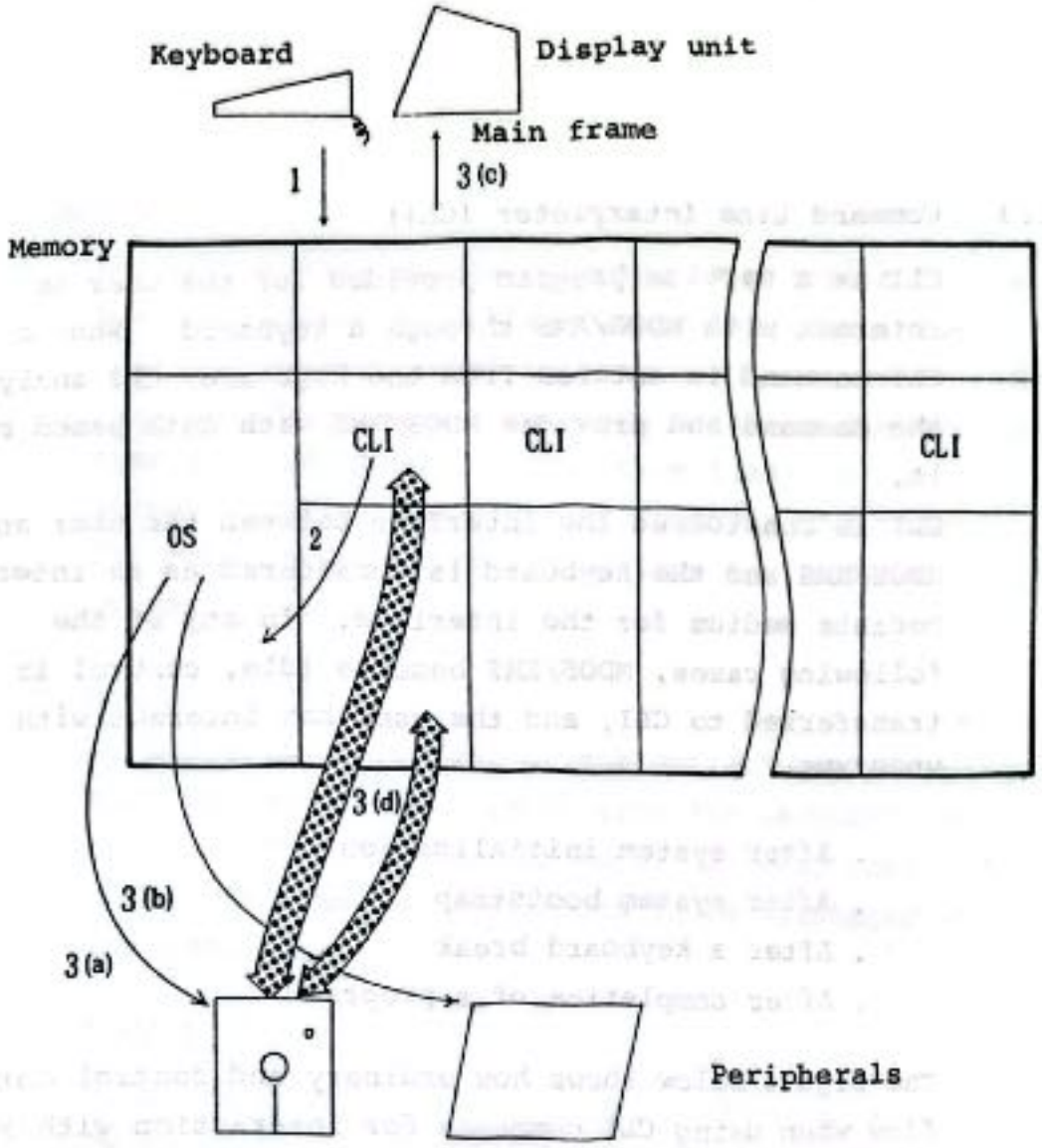
1.3 Command Line Interpreter (CLI)

CLI is a service program provided for the user to interact with MDOS/XMS through a keyboard. When a CLI command is entered from the keyboard, CLI analyzes the command and provides MDOS/XMS with data based on it.

CLI is considered the interface between the user and MDOS/XMS and the keyboard is considered as an intermediate medium for the interface. In any of the following cases, MDOS/XMS becomes idle, control is transferred to CLI, and the user can interact with MDOS/XMS:

- After system initialization
- After system bootstrap
- After a keyboard break
- After completion of a program

The figure below shows how ordinary and control data flow when using CLI commands for interaction with MDOS/XMS.



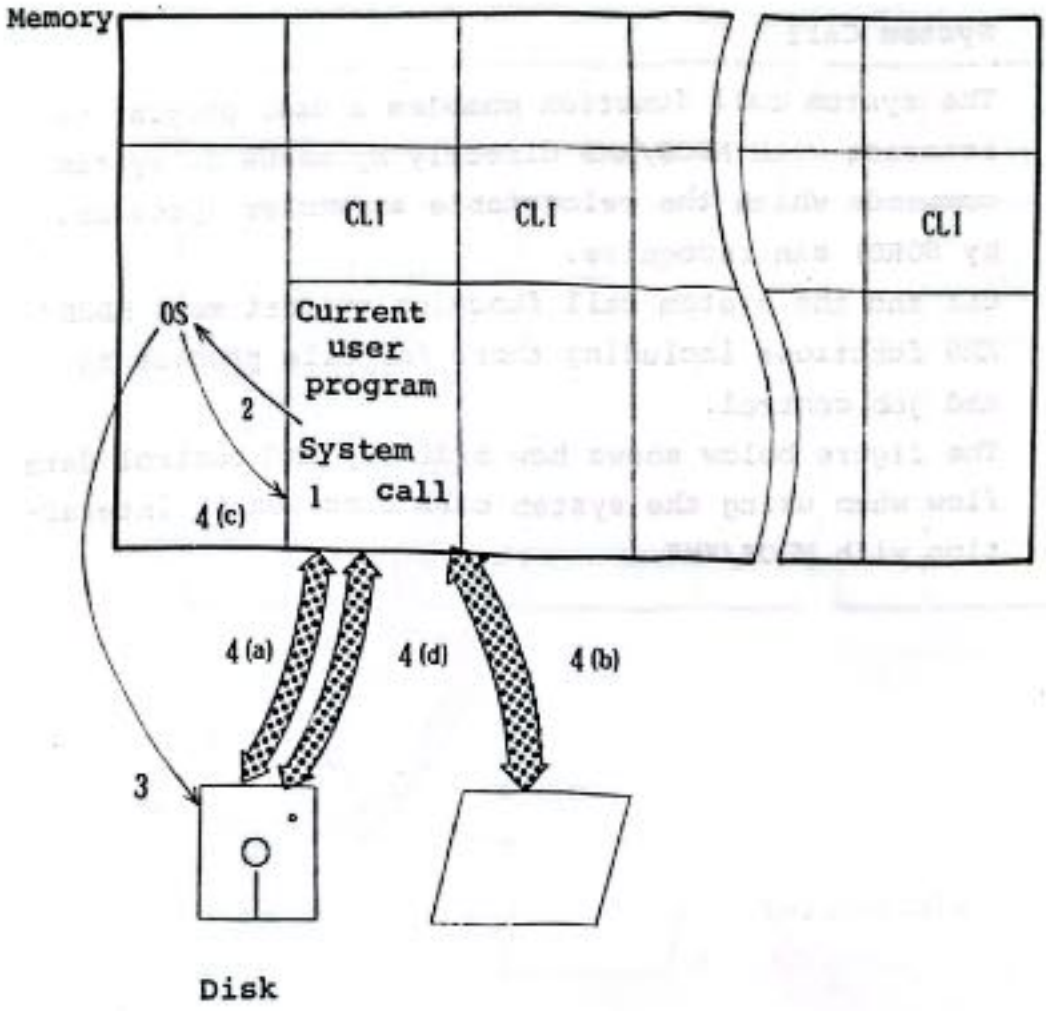
CLI and user programs

1. A CLI command is entered from the keyboard.
2. The CLI command is analyzed, and the required instruction is sent to MDOS/XMS or the required system maintenance is made.
3. (a) MDOS/XMS performs the required I/O operation on the required one or more disk drive units.
   (b) MDOS/XMS performs the required I/O operation between the required peripheral or disk drive unit, or between the required disk drive units.
   (c) The system outputs system data to the console.
   (d) MDOS/XMS executes the required program.

Fig. CLI mode

1.4 System Call

The system call function enables a user program to interact with MDOS/XMS directly by means of system commands which the relocatable assembler (provided by SORD) can recognize.
CLI and the system call function support most MDOS/XMS functions including those for file processing and job control.
The figure below shows how ordinary and control data flow when using the system call function in interaction with MDOS/XMS.



1. The current user program makes a system call.
2. Control is jumped to the MDOS/XMS monitor.
3. MDOS/XMS references the required control data on the required disk.
4. (a) The required I/O operation is made between the required disk and memory.
   (b) The required I/O operation is made between the required peripheral and memory.
   (c) The required system data is transferred.
   (d) The required program is chained.

Fig. Program mode

1.5 Files

The user can access any program, data, and I/O device easily by using system call because each of them are defined as a file.
Any of the following three types of file can be defined as a physical device:

- Disk file
- Memory file
- I/O file

Disk files

Disk files are stored on a disk in either of the following formats:

(1) Sequential file
A sequential file consists of disk blocks linked by link bytes and is controlled in units of a block. Therefore, when accessing data at the 10th block of a file, MDOS/XMS accesses the specified disk 10 times from the first block of the file to search for the required link byte.

(2) Contiguous file
A contiguous file consists of physically continuous disk blocks on a disk and MDOS/XMS can access any data in a file by just one access to the required disk.

Memory files

A memory file is created on a unused memory bank for temporary use and it is destroyed when the system power is turned off.
To use memory more efficiently, MDOS/XMS allows the user to handle a free memory bank as a file space in the same manner as for a disk. A memory file can be specified and accessed in exactly the same format as for a disk file, and the prescribed directory name in used to identify a memory file.

I/O files

The user can access any I/O device in the same manner as for a disk file because it is controlled as a file.
MDOS/XMS assigns a fixed file name to each of the following I/O devices, so the user can open, close, and access these I/O devices by using their file names.

- SINA RS232C A channel input
- SOUTA RS232C A channel output
- SINB RS232C B channel input
- SOUTB RS232C B channel output
- SINC RS232C C channel input
- SOUTC RS232C C channel output
- SIND RS232C D channel input
- SOUTD RS232C D channel output
- POUT Parallel port output
- KB Keyboard input
- CRT Display output
- SNSKY Sense key input
- LPEN Light pen input

1.6 Concept of Multijob Processing

When an I/O device is used in single job processing, the CPU cannot be used efficiently because the processing speed of an I/O device is always much lower than the CPU's and the CPU synchronizes with the I/O device in the processing speed.
In multijob processing, however, the CPU is allowed to execute a job while I/O operations for another job are being done. When the I/O operations are completed, the CPU is interrupted and executes the latter job again. Processing like this enables more than one job to be processed concurrently and allows the CPU to be used more efficiently.

1.7 Disk File Types

Disk files are largely divided into the following six types:

- System files
- CLI command files
- System program files
- System utility program files
- Application package program files
- User program files

The first five types of program files are provided by SORD.

System files

System files are used for OS to control its functions and the names of these files on a disk are registered in the system directory.
The following system files are provided on a disk:

- BOOT.SYS (Bootstraps OS.)
- DISK.SYS (Contains disk control information.)
- SYS.DIR (System directory)
- IO.DLR (I/O file directory)
- MEMO.DIR (Memory file directory)
- MDOS.SYS (MDOS/XMS operating system file)

CLI command file

Various CLI command files are provided for interaction with MDOS/XMS through the console.

System program files

The following system program files are provided:

- BASIC (Compiler)
- BASIC-II
- SBASIC
- Relocatable assembler
- Editor
- Debugger
- Relocatable loader
- Library file editor
- FORTRAN
- COBOL
- PIPS (SORD original simplified language)
- SGL (SORD Graphic Language)

System utility program files

The following system utility program files are provided:

- . INIT (Floppy disk formatter)
- . SIFT (1 MB floppy disk convertor)
- . SST (Program for file transfer between SORD different modules)
- . MSORT (Multi-key sort program)
- . TAPFER (Paper tape reader/puncher support program)
- . CMTDL (Program for dumping and loading data to 'CMT-10')
- . MTDL (Program for dumping and loading data to 'MTU-80')
- . CMTP3 (Program for dumping and loading disk files to 'CMT-10')
- . MTBK (Program for dumping and loading disk files to 'MTU-80')
- . MDUMP (Program for dumping files to a floppy disk)
- . MLOAD (Program for loading files from a floppy disk)
- . ADUMP (Program for dumping disk files to a floppy disk)
- . ALOAD (Program for loading files from a floppy disk)

Application program package files

The following application program package is provided by SORD:

. Word processor for the English language

User programs

The user can develop programs by using various SORD system programs and store them on disks. Such programs are called user programs.

CHAPTER 2 FILE MANAGEMENT

2.1 Memory Files

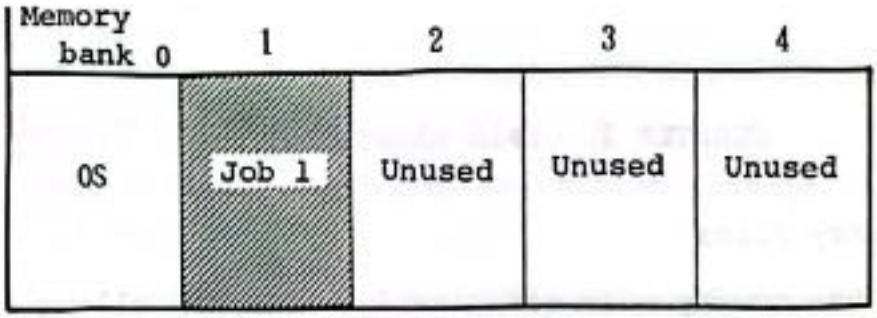
To use memory more efficiently, MDOS/XMS allows the user to use a unused memory bank as a very high-speed external storage device. A file created on a memory bank is called a memory file, which is controlled by MDOS/XMS.
A memory file can be handled in exactly the same manner as for a disk file and specified by using the directory name MEMO.DIR. MEMO.DIR is a fixed name and cannot be used as a directory name for a disk.

Example

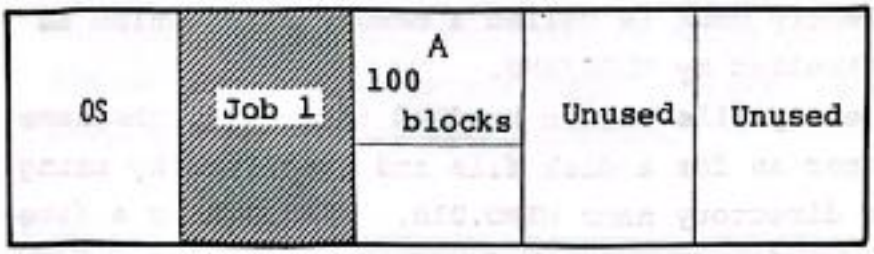
```
CREATE [MEMO] DATA/C 80X500
```

Creates the continuous file DATA at 80 bytes by 500 records as a memory file.

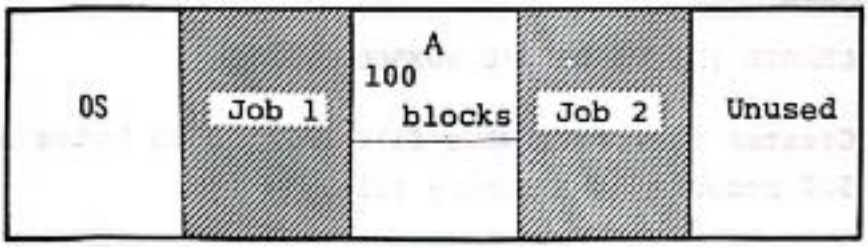
A memory file is controlled also in units of 512 bytes per block like a disk file and 128 blocks are available in one 64K byte memory bank. A file can be created over two or more memory banks so long as a memory capacity allows it. However, a bank cannot be used for a job and a memory file at the same time. That is, if a bank is used for a job, the bank cannot be used as a file space, and if a bank is used for a memory file, the bank cannot be used for a job. When reservation of a memory bank is specified for use as a memory file space, MDOS/XMS searches for a free memory bank and reserves it if available. Therefore, the user can specify any memory file without checking the physical location of a free memory bank.



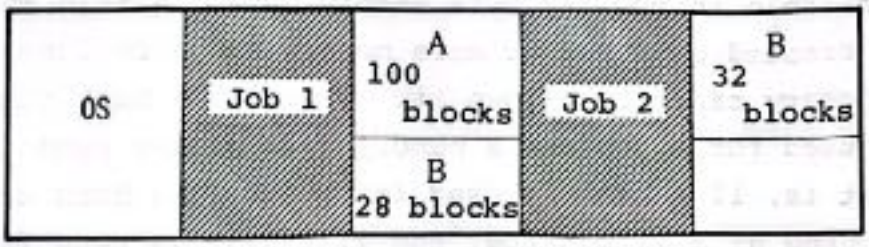
① Job 1 is activated.



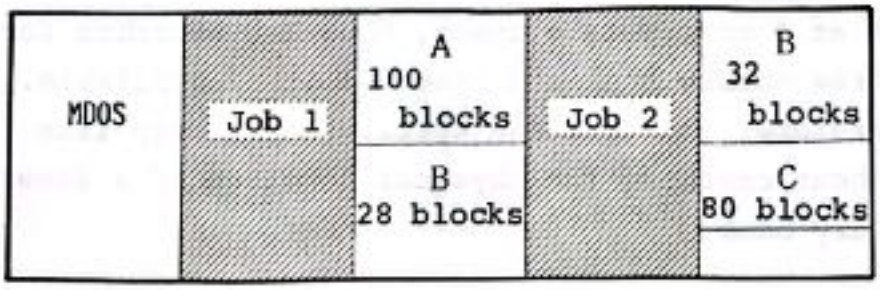
② Job 1 specifies 'CREATE [MEMO] A/C 512X100'.



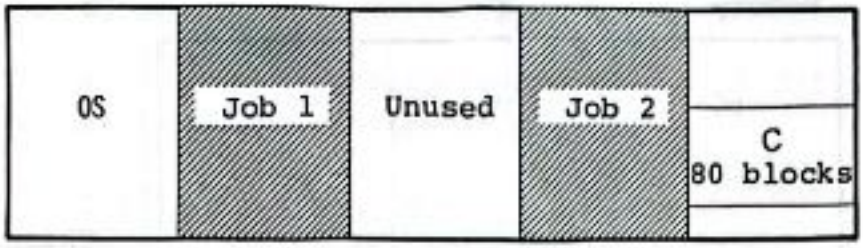
③ Job 2 is activated.



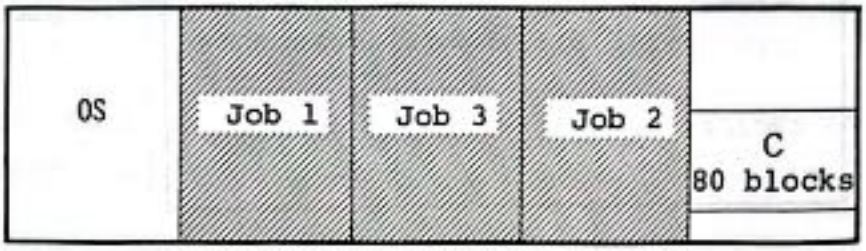
④ Job 1 specifies 'CREATE [MEMO] B/C 512X60'.



⑤ Job 2 specifies 'CREATE [MEMO] C/C 512X80'.

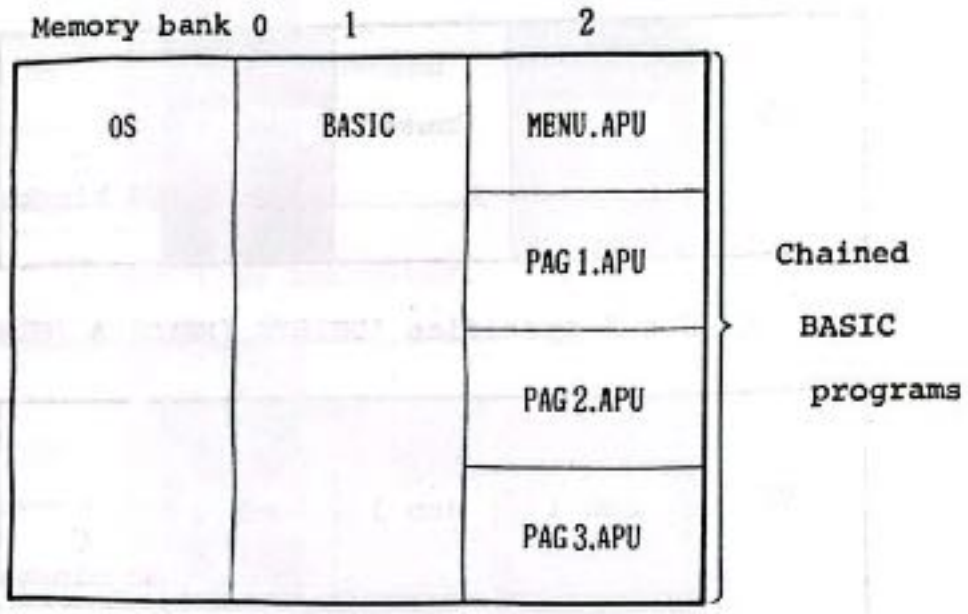


⑥ Job 1 specifies 'DELETE [MEMO] A [MEMO] B'.



⑦ Job 3 is activated.

A memory file can be accessed about 10 times faster than a file on a winchester fixed disk is, so this memory file function is useful in high-speed sampling of a medium length of data. Furthermore, the time it takes to chain BASIC programs can be reduced by storing them as memory files.



Since any memory file is destroyed when the system power is turned off as written earlier, memory files necessary for later use must be copied to disks.
The memory file directory is reserved in the MDOS/XMS memory area and themaximum number of memory files which can be registered is set at system generation.

2.2 Disk Files

MDOS/XMS supports the following five disk drive units for the M243 mark series computer:

- 1 MB mini-floppy disk drive unit
- Standard 1 MB floppy disk drive unit
- 8 MB winchester fixed disk drive unit
- 10 MB winchester fixed disk drive unit
- 20 MB winchester fixed disk drive unit

The following table shows the specifications of disks supported by MDOS/XMS:

Table Supported disk specifications

Specifications	1 MB mini-floppy disk FA	Standard 1 MB floppy disk FA	10 MB Winchester fixed disk WA	20 MB Winchester fixed disk WB	8 MB Winchester fixed disk WC
Number of cylinders per volume	77	75	240	240	153
Number of tracks per cylinder	2	2	4	8	6
Number of sectors per track	15	15	20	20	17
Number of bytes per sector	512	512	512	512	512
Total number of blocks	2250	2250	19200	38400	15606
User capacity					
When MDOS/XMS is included	1972	1972	18792	37872	15240
When MDOS/XMS is not included	2100	2100	18920	38000	15368
Maximum number of files which can be registered	240	240	640	960	544
Maximum number of file names which can be registered in the system directory	64	64	64	64	64

To support these disk drive units in the same manner, MDOS/XMS assigns a logical disk drive unit name and a logical disk drive unit number to each drive unit. To specify a disk type, the user must use its logical disk drive unit name or number.
Table shows logical disk drive unit names which can be specified in CLI commands and logical disk drive unit nubers which can be specified in system call.

Table Logical disk drive unit names and numbers

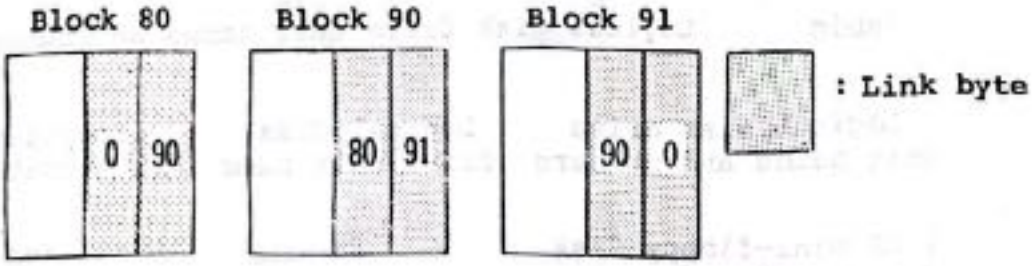
Logical disk drive unit names and numbers	Logical disk drive unit name	Logical disk drive unit number
1 MB mini-floppy disk drive unit	FA	'02
Standard 1 MB floppy disk drive unit	FA	'02
10 MB winchester fixed disk drive unit	WA	'03
20 MB winchester fixed disk drive unit	WB	'13
8 MB winchester fixed disk drive unit	WC	'23

Disk file types

There are two types of disk file as follows:

(1) Sequential file

In a sequential file, the last eight bytes of a block of 512 bytes is used as a pointer to indicate the preceding and succeeding block addresses.
The last eight bytes of each block are called link bytes.
Pointers are controlled by MDOS/XMS and the user does not need to consider them.

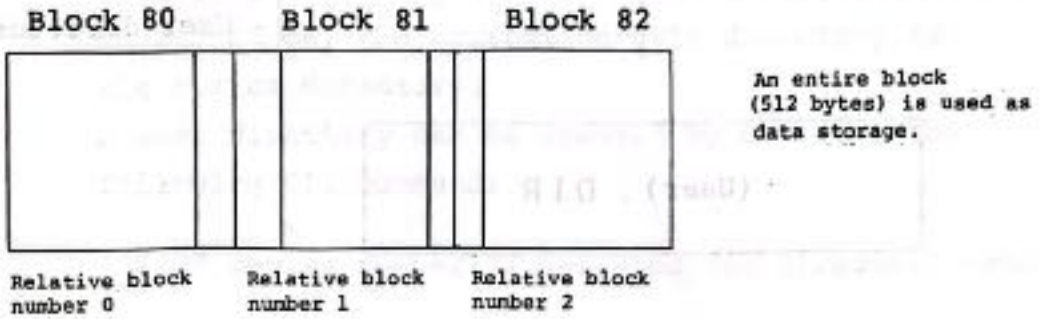


Sequential file structure

Blocks of a sequential file need not be arranged sequentially on a disk, that is, they need not be physically continuous on a disk, because they are linked by link bytes. Therefore, a sequential file can be extended easily by using free spaces at random on a disk. When accessing a sequential file, it must be done sequentially from the first block.

(2) Contiguous file

A contiguous file can be accessed at random by specifying the required record. Contiguous file blocks must be physically continuous on a disk. Therefore, a contiguous file cannot be extended unlike a sequential file.


Block 80
Block 81
Block 82
An entire block (512 bytes) is used as data storage.
Relative block number 0
Relative block number 1
Relative block number 2

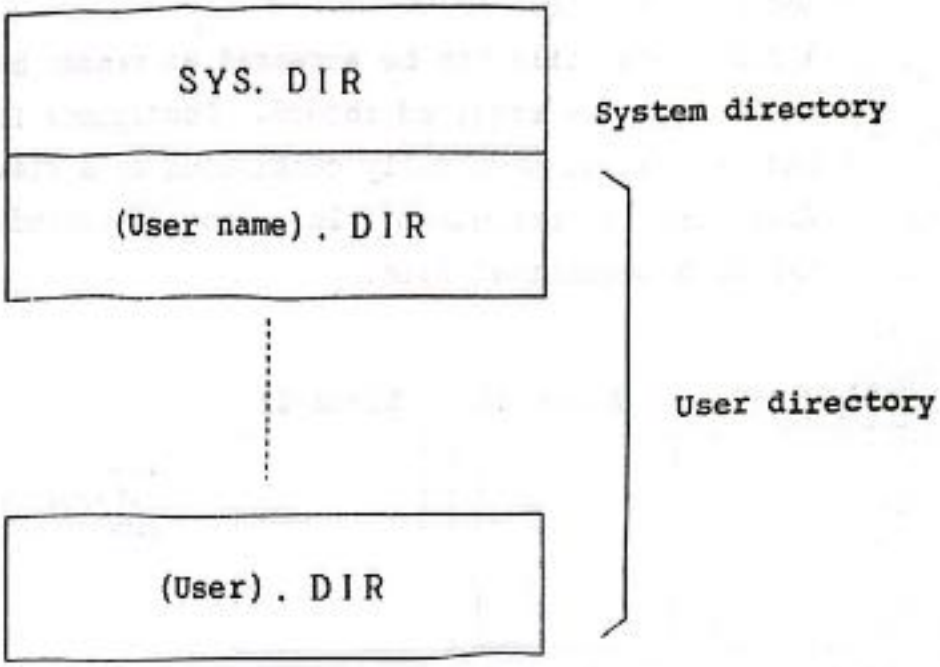
Contiguous file structure

2.3 Directories

Directories are a kind of file ledger used to control files on a disk and each disk is always provided with at least one directory. In order for the user to be able to control files and search for a file more easily, MDOS/XMS allows more than one directory to exist on a disk.

The following two types of directory can exist on a disk:

- System directory
- User directory



System directory

On a disk, only one system directory can exist and its filename is SYS.DIR. The names of other files including user directory files, CLI command files, and system utility files are registered in the system directory.

User directories

A user directory is created by the user and its directory filename is specified also by the user.

File reference

Generally, when specifying a file, the directory in which the name of the file is registered must be specified at the same time. If no directory is specified, the current default directory is used. The current default directory means a directory currently opened. When the system power is turned on, the current default directory is the system directory.
A user directory can be created by entering the following CLI command:

DIRT device number: [user-specific directory name]

If a user directory is created, the directory is now the current default directory. As a result, any file registered in the user directory can be referenced only with its file name without specifying the directory name. The user directory remains the current default directory until the next user directory is created. (More than one user directory cannot be created at the same time.)
The same file name can be used on a disk so long as it is registered in different user directories.

User directory registration and deletion

To register a user directory name, a CLI command must be specified as follows:

CDIR device number: [directory name] number of files

As a result, the user directory name is registered in the system directory of the specified disk. If the directory name already exists, an error message is output.
To delete a user directory name from the system directory, a CLI command must be specified as follows:

```
DDIR device number: [directory name]
```

As a result, the user directory name is deleted from the system directory of the specified disk. Furthermore, all files registered in the user directory are also deleted from the disk regardless of their attributes.

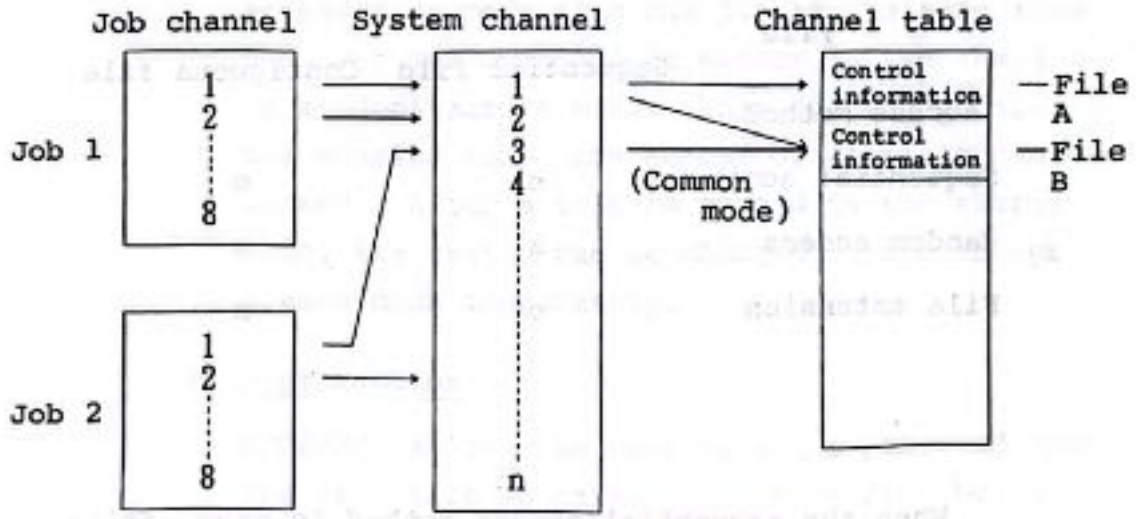
Directory protection

To prevent loss of files which will occur if a directory on a disk is destroyed, MDOS/XMS provides the directory back-up function to update back-up directories each time their original directories are updated. Therefore, even if a directory is destroyed, the user can recover the directory by transferring the contents of the back-up directory to its area. Start and stop of the directory back-up function must be specified by the user, but the location of the back-up directory need not be specified because it is located in the fixed area of the system. The directory back-up function does not replace only a destroyed directory, but replaces all directories at the same time.
This back-up function can be used in units of a disk, so the user can select a disk for which the function is to be used. Furthermore, the user can check whether the required directory is being recovered normally or not as required.

2.4 File Access Methods

Opening files

To access a file, the file must be opened beforehand to link it to a channel. The accessing of the file is controlled by the channel and the user accesses the file by the channel number, not by the file name, after the file was opened. More than one file cannot be linked to a channel. Up to eight channels from No.1 to No.8 can be assigned to one job.
Figure shows the relationship between job channels and system channels in opening of files.



Accessing files

MDOS/XMS accesses a disk file by either of the following methods:

(1) Sequential access
MDOS/XMS accesses the file sequentially from the first block.

(2) Random access
MDOS/XMS divides the file in units of a record and accesses a specified record at random. The following table shows the relationship between file formats and access methods:

File / Access method	Sequential file	Contiguous file
Sequential access	o	o
Random access	x	o
File extension	o	x

When the sequential access method is used, files are accessed sequentially according to the file pointer.

In the sequential access method, the file pointer means the byte pointer indicating a file to be accessed next and usually indicates the first byte (Byte 0) of the file when the file is opened ($OPEN). When a file is opened in the additional mode ($APEND), the file pointer indicates the EOD mark of the file and moves each time an access is made.

Closing files

After use of a file is completed, the file is unlinked from the channel.

File sharing mode

MDOS/XMS file management allows a file to be accessed by more than one job at the same time. The user can open a file either in the sharing or monopolization mode. When opening a file in the sharing mode, any number of channels can be linked. After a file is opened in the sharing mode, the system can be changed to the monopolization mode temporarily.

File protection

MDOS/XMS allows the user to set a password code for each file so as to prevent it from being referenced by unauthorized users. If a correct password code is not entered, the file cannot be referenced.

A password can be specified with four bytes of code, and if it is specified with NULL codes, the file can be referenced by any user.

CHAPTER 3 CONSOLE CONTROL

3.1 Keyboard

Figure 3.1 shows the keyboard arrangement.

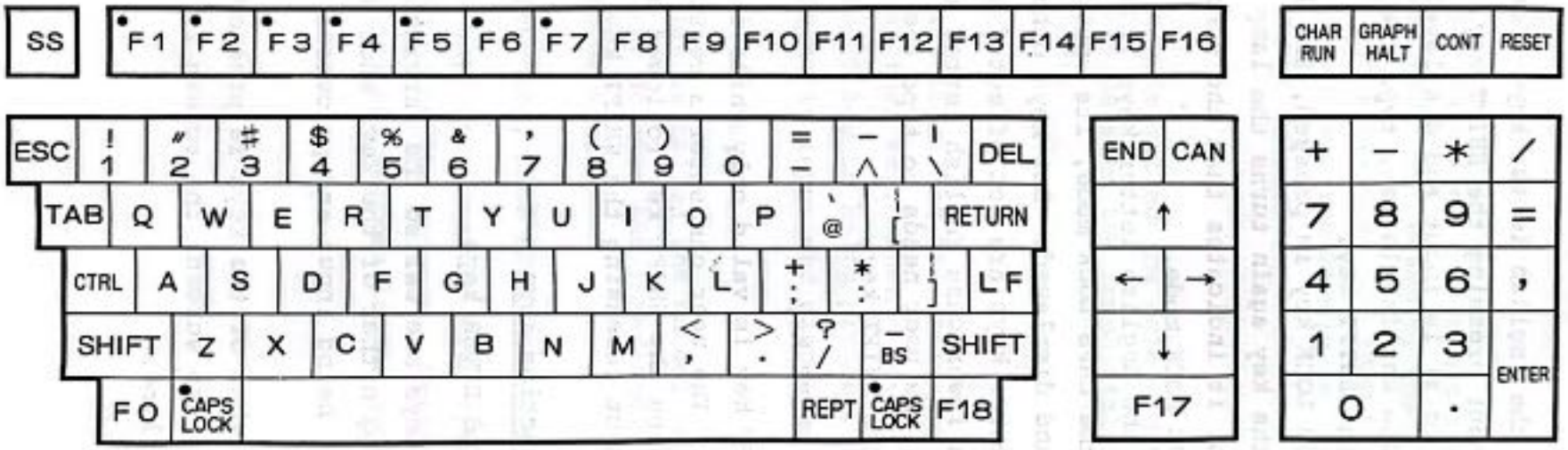

SS
F1 F2 F3 F4 F5 F6 F7 F8 F9 F10 F11 F12 F13 F14 F15 F16
CHAR RUN
GRAPH HALT
CONT
RESET
ESC ! 1 " 2 # 3 $ 4 % 5 & 6 ' 7 (8) 9 0 = - ~ ^ | \ DEL
TAB Q W E R T Y U I O P ` @ { [RETURN
CTRL A S D F G H J K L + ; * : }] LF
SHIFT Z X C V B N M < , > . ? / _ BS SHIFT
F0 CAPS LOCK REPT CAPS LOCK F18
END CAN
↑
← →
↓
F17
+ - * /
7 8 9 =
4 5 6 ,
1 2 3 ENTER
0 .

Fig. 3.1 ASCII keyboard arrangement

If any of the English letter keys from A to Z is typed without pressing the SHIFT key, its lower-case letter (a to z) is input and displayed. To enter a upper-case English letter, type the required key, pressing the SHIFT key.
If the CAPS LOCK key is pressed, the lamp turns on. Pressing the key again turns the lamp off. When the lamp is on, it indicates that the console operates in the CAPS LOCK mode.
If any of the English letter keys from A to Z is typed in the CAPS LOCK mode, its upper-case letter is input and displayed. But keys other than these English letter keys are not affected by this mode. To enter a lower-case English letter in the CAPS LOCK mode, the user needs to type the required key, pressing the SHIFT key.

Using the SHIFT key

The SHIFT key is valid only while it is being pressed. The user can enter a required key code by pressing the SHIFT key followed by the required key without releasing the SHIFT key.

Special function keys

RETURN and ENTER keys

These keys are called the carriage return keys. Pressing either of the keys ends a line on which data is being input and processes the line of data.
When either of the keys is pressed on the last line of the screen, the screen is scrolled up by one line.

REPT key

This key is called the repeat key and used with another key. For example, when the REPT key and A key are pressed at the same time, the A key code is entered repeatedly so long as the REPT key is not released.

DEL key

This key is called the delete key. Pressing this key deletes the immediately preceding character.

SHIFT key

This key is called the shift key and usually used with another key. When a key is pressed while the SHIFT key is being pressed, the upper-case character is input and displayed.

CAN key (ten key)

This key is called the cancel key. Pressing this key deletes the immediately preceding character string.

ESC, CTRL, TAB, LINE FEED, BS, and END keys

These keys are provided with special functions, depending on the program. Details on the special functions are written in the appropriate sections.

System command keys

The following keys and key combinations operate as system command keys:

SHIFT and CHAR RUN keys

The character screen can be turned on and off by using the SHIFT and CHAR RUN keys together. That is, if these keys are pressed at the same time when the character screen is displayed, the screen is cleared. And if they are pressed again at the same time, the character screen is displayed again.

SHIFT and GRAPH HALT keys

The graphic screen can be turned on and off by using the SHIFT and GRAPH HALT keys together. That is, if these keys are pressed at the same time when the graphic screen is displayed, the screen is cleared. And if they are pressed again at the same time, the graphic screen is displayed again.

CHAR RUN and GRAPH CONT keys

These keys are used to execute or continue a BASIC program. Pressing the CHAR RUN key has exactly the same functions as when RUN is typed and then the RETURN key is pressed. Pressing the GRAPH CONT key has exactly the same function as when CONT is typed and then the RETURN key is pressed.

HALT key, or SHIFT and RESET keys

Usually, these key and key combination are both used to terminate execution of a program. They are sometimes provided with special functions by programs.

System command keys with the CTRL key

CTRL and 0 keys

When these keys are pressed at the same time, display is stopped, and when they are pressed again at the same time, the screen is displayed again. This operation just stops or restarts display on the screen, and even if display is stopped, processing is continued.

CTRL and S keys

When these keys are pressed at the same time, display is stopped and the related job is also stopped.

CTRL and Q keys

Pressing these keys at the same time restarts the display and job stopped by pressing the CTRL and S keys.

CTRL and R keys

If these keys are pressed at the same time during data entry, data entered up to the point is displayed again after a line feed. The user can then continue data entry.

CTRL and Z keys

If these keys are pressed at the same time when the console is handled as a file, the file is closed.
The CTRL key is provided with a special function, which depends on the program. Details on CTRL key functions are written in the appropriate sections.

Function keys

The user can provide a function key with a special function by entering a character string with the FUNC command. And, by using the function key with SHIFT key, it can be provided with another function which is different from its original function. This applies for all function keys, thus the user can assign total 42 functions.

For details on how to provide each function key with a special function, see Section 7.15.

The following character strings are assigned to individual function keys for the standard keyboard:

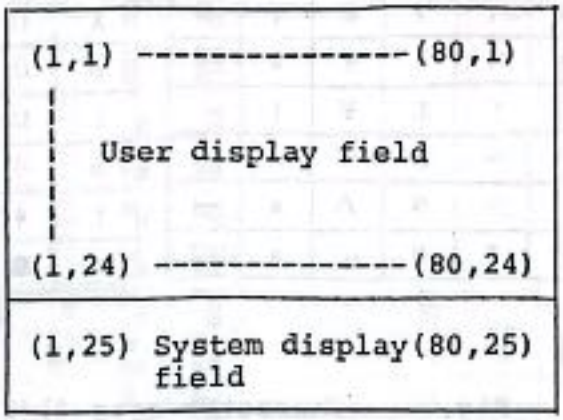
```
#List
      ********************************************************
F00   :<'17>:                  f00   :<'17>:
F01   : FUNC CLI<'00>:         f01   :FUNC :
F02   :FUNC BASIC<'00>:        f02   ::
F03   :CDIR :                  f03   :DDIR :
F04   :DIRT :                  f04   ::
F05   :GOSUB:                  f05   :LEN(:
F06   :RETURN<'0D>:            f06   :ATN(:
F07   :INPUT:                  f07   :USING:
F08   :THEN:                   f08   :END<'0D>:
F09   :GOTO:                   f09   :LN(:
F10   :PRINT:                  f10   :TAN(:
F11   :EDIT:                   f11   :RECORD:
F12   :LIST<'0D>:              f12   :EXP(:
F13   :SAVE:                   f13   :CLOSE:
F14   :OLD:                    f14   :DELETE:
F15   :BASIC<'0D>:             f15   :ABS(:
F16   ::                       f16   ::
F17   :<'0D>:                  f17   :<'0D>:
F18   ::                       f18   ::
F19   : :                      f19   : :
F20   : :                      f20   : :
[F1]  ::                       [f1]::
[F2]  ::                       [f2]::
[F3]  ::                       [f3]::
[F4]  ::                       [f4]::
      ********************************************************
```

In the above figure, F represents the case that the function key is not used with the SHIFT key and f represents the case that the function key is used with the SHIFT key.

Sense switch keys

Seven function keys from F01 to F07 are used also as sense switch keys.
To use these keys as sense switch keys, the user needs to use the SS key at the left-hand side of the function key row at the same time. For example, to specify the sense switch key 1, the SS and F1 keys should be pressed at the same time. When both keys are pressed, the lamp at the upper left-hand side of the F1 key turns on, which indicates that the sense switch key 1 has been pressed.

3.2 Display Unit



Character sets

MDOS/XMS supports six character sets for console display: one G0 set (characters represented by Codes '20' to '7E'), three G1 sets (characters and graphic symbols represented by Codes 'A0' to 'FF'), and the other two are C0 and C1 (control codes represented by Codes '00' to '1F' and '80' and '9F' respectively).

Control codes
G0 set

Low order 4 bits \ High order 4 bits	0	1
0		D_L
1	S_H	D_1
2	S_X	D_2
3	E_X	D_3
4	E_T	D_4
5	E_Q	N_K
6	A_K	S_Y
7	B_L	E_B
8	B_S	C_N
9	H_T	E_M
A	L_F	S_B
B	V_T	E_C
C	F_F	F_S
D	C_R	G_S
E	S_O	R_S
F	S_I	U_S

Alphanumeric characters
G0 set

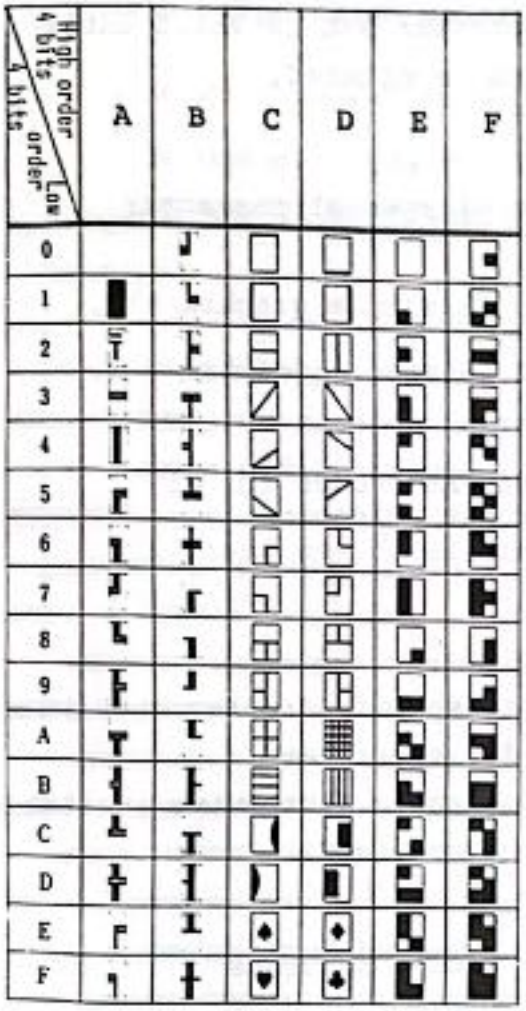
Low order 4 bits \ High order 4 bits	2	3	4	5	6	7
0	(SPACE)	0	@	P	`	p
1	!	1	A	Q	a	q
2	"	2	B	R	b	r
3	#	3	C	S	c	s
4	$	4	D	T	d	t
5	%	5	E	U	e	u
6	&	6	F	V	f	v
7	'	7	G	W	g	w
8	(	8	H	X	h	x
9	)	9	I	Y	i	y
A	*	:	J	Z	j	z
B	+	;	K	[	k	{
C	,	<	L	¥	l	|
D	-	=	M	]	m	}
E	.	>	N	^	n	~
F	/	?	O	_	o	(DEL)

Special characters
G1 set

Low order 4 bits \ High order 4 bits	A	B	C	D	E	F
0						
1						
2						
3						
4						
5						
6						
7						
8						
9						
A						
B						
C						
D						
E						
F						

Fig. Character sets (1/2)

Simple graphic - 1
Gl set



Simple graphic - 2
Gl set

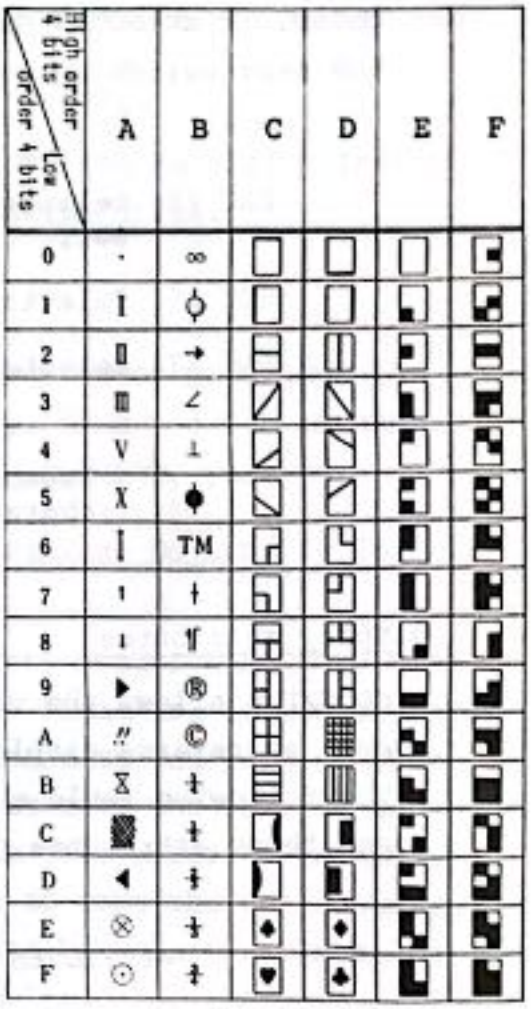


Fig. Character sets (2/2)

Display

Since there are three G1 character sets, MDOS/XMS switches them in display by using escape sequence codes. A selected G1 character set is valid until the next switching command is entered.

ESC)5	Selects the G1 special character set.
ESC)6	Selects the G1 simple graphic - 1.
ESC)7	Selects the G1 simple graphic - 2.

Fig. Escape sequence codes for G1 character set selection

Display attributes

MDOS/XMS allows the user to specify display attributes, such as reverse display, for characters.
The following table shows display attributes available and these attributes can be combined in use:

Attribute	Escape sequence
No attribute	CSI 0 m
Highlighting	CSI 1 m
Underline	CSI 4 m
Blinking	CSI 5 m
Reverse	CSI 7 m
Color display	CSI 30 m CSI 37 m

CSI indicates code '9B' (hex.).

CHAPTER 4 MULTIJOB PROCESSING

4.1 Purposes of Multijob Processing

Main purposes of multijob processing are as follows:

. To use the CPU more efficiently.

. To use external storage devices more efficiently by two or more users' sharing files.

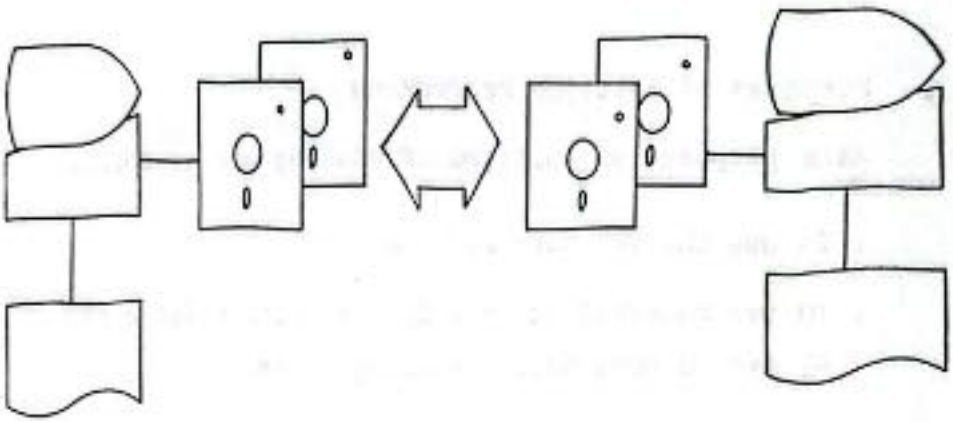
Efficient use of the CPU

Multijob processing allows two or more jobs to be processed concurrently, thus the CPU is used more efficiently.

Efficient use of external storage devices by file sharing

Multijob processing is very useful when more than one user is sharing data. That is, if two or more users need to use the same data and each of them uses one system, they must have the same data individually. And if the data needs to be updated, these users are required to copy the data between media or transfer the data between their systems by connecting them by cables.
In multijob processing, more than one terminal is connected to one computer and more than one user can use the CPU at the same time. Data to be shared is stored on a disk and each user can reference or update the data as required. Thus, the user can construct a system more economically and manage data more easily.

Single job processing



Data copy between floppy disks or connection of systems by the RS232C standard cable

Multijob processing

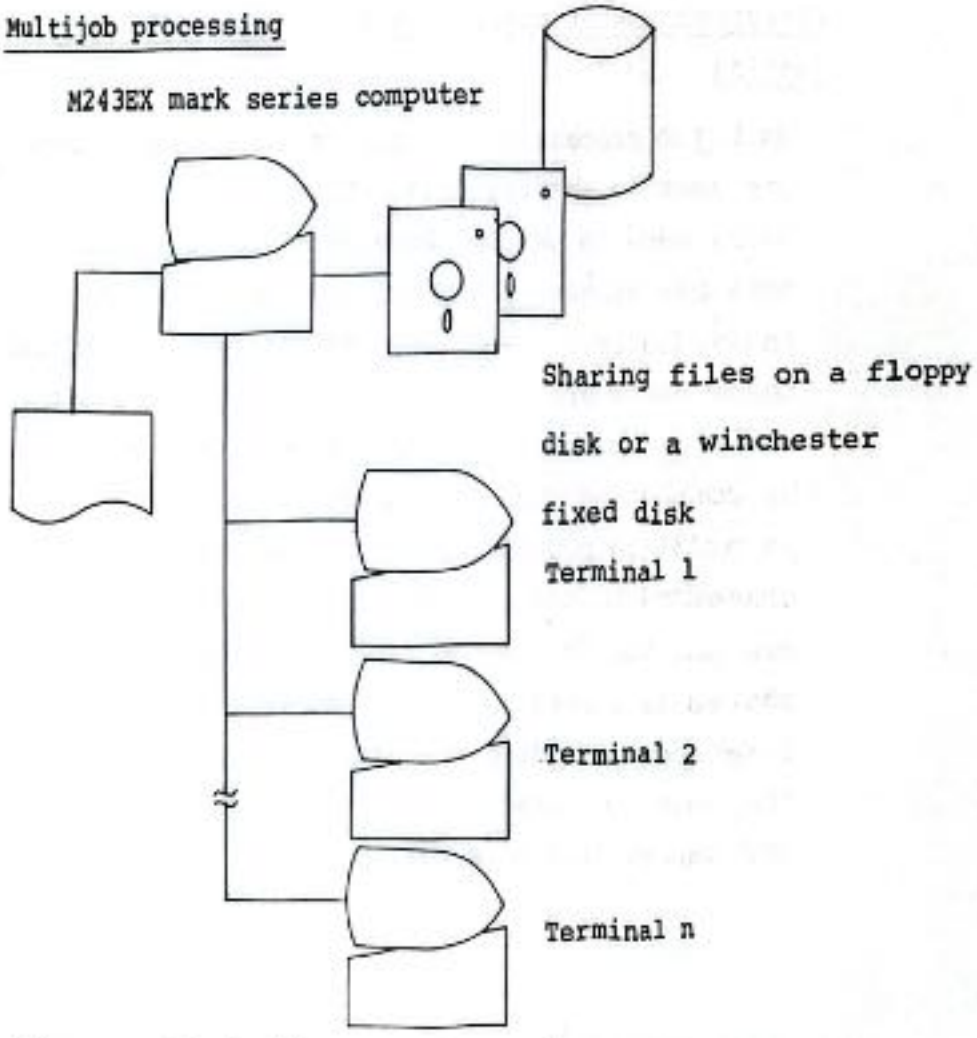
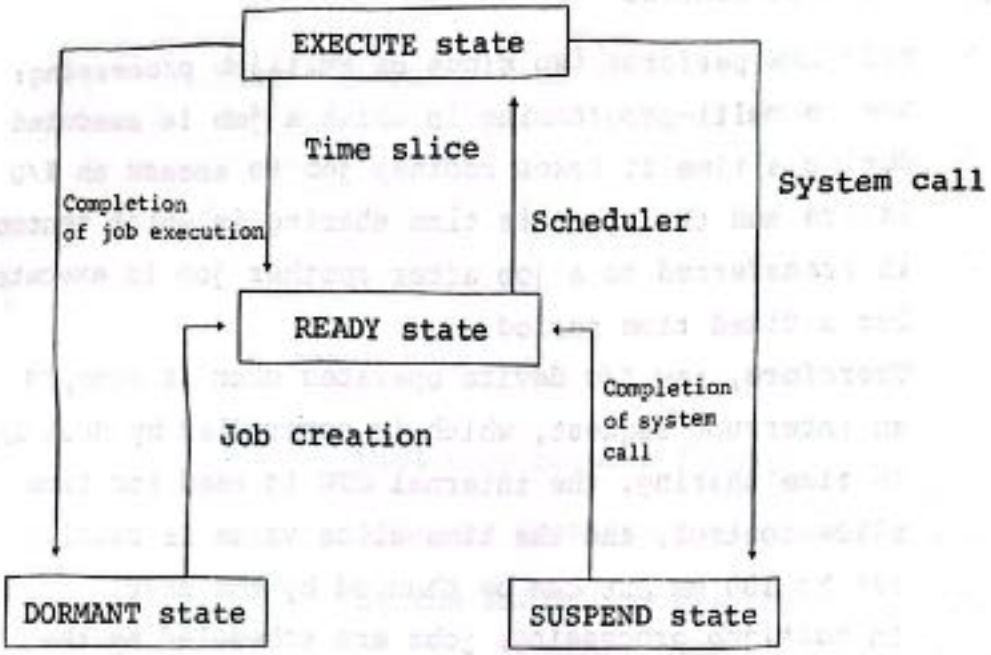


Fig. Single job processing and multijob processing

4.2 Multijob Control

MDOS/XMS performs two kinds of multijob processing: one is multi-programming in which a job is executed during a time it takes another job to access an I/O device and the other is time sharing in which control is transferred to a job after another job is executed for a fixed time period.
Therefore, any I/O device operates when it accepts an interrupt request, which is controlled by MDOS/XMS.
In time sharing, the internal CTC is used for time slice control, and the time slice value is usually set to 100 ms but can be changed by the user.
In multijob processing, jobs are scheduled by the job scheduler according to the round robin method. That is, a job which has received a CPU service for a fixed time period (EXECUTE state) is entered at the last of the job execution queue. The job cannot receive a CPU service again until all other jobs in the queue receive CPU services. However, if an interrupt request is sent, the required job can received a CPU service with top priority even if it is in the SUSPEND state.



1) EXECUTE state

Indicates that a job is currently receiving a CPU service.

2) READY state

Indicates that a job is ready to receive a CPU service.

3) SUSPEND state

Indicates that a job cannot receive a CPU service temporarily because it is waiting for completion of I/O operations made by system call.

4) DORMANT state

Indicates that a job is not ready to receive a CPU service or execution of a job has been completed.

Fig. 5.2 Job state transitions

4.3 Job Types and Activating Jobs

There are two types of jobs:

. Foreground jobs
. Background jobs

Foreground jobs

Any foreground job specifies system call (such as $DSPLN or $ACELN) in it for the console which has been used to activate the job. It requires communication with an operator through the console. Therefore, a console must be allocated for each foreground job.
A foreground job is activated by entering the job file name in response to the READY prompt displayed on the console screen. The activated job monopolizes the console. So background jobs should be activated prior to foreground jobs.

Background jobs

Any background job does not use system call for the console which has been used to activate the job. It does not require communication with an operator through the console, because no communication with an operator is needed or data to be exchanged between the job and the operator can be prepared beforehand. A foreground job can be activated by typing JOB and entering the job file name in response to the READY prompt.

Example:

```
READY
JOB SHORI <RETURN>: Specifies a background job.
```

After a job is activated, the operator needs to wait for the next CLI command, that is, for the job to be created. If it is impossible to create the job because memory is insufficient, an error message is displayed when activation of the job is specified and the job is not executed.

Specification of more than one background jobs

MDOS/XMS allows more than one job to be created on the same memory bank by specifying job file names as follows:

```
JOB XXXX;XXXX1;XXXX2;----;XXXXN
```

If job file names are specified as written above, the job XXXX1 is executed after execution of the job XXXX is completed. In the same manner, the job XXXX2 is executed after execution of the job XXXX1 is completed.

Alternate console file function for backgorund jobs

MDOS/XMS allows the user to create alternate files for keyboards and display units. If such a file is created for a keyboard and a system call requests data entry from the keyboard, data can be read from the file and transferred to the job. If an alternate file is created for a display unit and a system call requests data output to the display unit, the data can be output to the file instead.

This function is useful when data to be entered from a keyboard can be prepared beforehand and when an output result should be referenced.
For this purpose, two channels, the user channels 7 (for input) and 8 (for output), are reserved to access input and output alternate files for keyboards and display units. User jobs cannot use these channels.
The following example shows how to compile a BASIC program as a background job:

Example: Procedure for compiling the program TEST.APU as a background job

Step 1 To compile the program TEST.APU, proceed as follows:

```
BASIC
  *OLD TEST
  *COMPILE
    FILE NAME:TEST
 READY
```

where the user enters the following data from the keyboard after the BASIC program is called in:

```
OLD TEST
COMPILE
TEST
```

The user must then create an alternate file and write the above data into the file, using the editor.
The alternate file name is KEY.IN.

```
EDIT
 *GWKEY.IN$ $
 *IOLD TEST
COMPILE
TEST
$ $
*P $GC$H$$
READY
```

Then, another alternate file for output to the display unit must be created with the name DSP.OUT.

```
CREATE DSP. OUT
```

Step 2 Specify the program as a background job as follows:

```
Job / I / O  KEY.IN   DSP.OUT  BASIC
```

Specifies an output file name.

Specifies an input file name.

Indicates that the execution result is to be sent to an output file.

Indicates that data is to be read in from an alternate file.

When only an input file is prepared

```
JOB/I KEY. IN BASIC
```

When only an output file is prepared

```
JOB/O DSP. OUT BASIC
```

When both an input file and an output file are to be specified, their names must be specified in the order of the input file name and output file name as written in the above example.

CHAPTER 5 MDOS/XMS BOOTSTRAP PROCEDURE

If MDOS/XMS is to be bootstrapped from a floppy disk and the system power is turned on, the following message is displayed:

"SET DISKETTE"

Insert the appropriate floppy disk into a unit. MDOS/XMS is loaded from the unit disk. If two or more floppy disks are inserted at the same time, MDOS/XMS is loaded from the disk recognized to be inserted first.
If MDOS/XMS is to be bootstrapped from the winchester fixed disk and the system power is turned on, the following is message indicating that MDOS/XMS is being bootstrapped is displayed:

"WAITING FOR DISK READY"

The user can specify from which device, a floppy disk drive unit or the winchester fixed disk drive unit, MDOS/XMS is to be bootstrapped, by using the DIP switch at the rear panel of the main frame. For details on how the DIP switch is to be set, see Appendix.
When loading of MDOS/XMS is completed, the following message (bootstrap message) is displayed on the main console and terminals:

"M24 MDOS/XMS Version 31A Released at 01-AUG-83"

These terminals should be turned on before bootstrap. If they have not been turned on before or during bootstrap, the above message is displayed when they are turned on.

The current time and date (day, month, year, hour, minute, and second) are displayed in the following format at the lower right-hand side of the system display field of the main console screen:

"DD-MM-YY HH:MM:SS"

If the displayed time or date is incorrect, the user can correct it by using the CLI TIME or DATE command respectively. If an error occurs during bootstrap, one of the following messages is displayed on the main console screen:

"NOT M24 DISKETTE!"

The user attempted to access a disk which is not supported for the M243EX computer.

"NOT OS DISKETTE!"

The disk does not contain MDOS/XMS. Turn the system power off, load the appropriate disk, and then bootstrap MDOS/XMS again.

"ERROR M"
"ERROR 1"
"ERROR 2"
"ERROR 3"

If one of the last four messages is displayed, contact a SORD personnel or a SORD representative.

Logging in.

The user is not allowed to execute a CLI command even if bootstrap has been completed and the bootstrap message is displayed. To execute a CLI command, the user must log into the system according to the following procedure:

. Press the RETURN key. The system displays the prompt ':::'.

. In response to this prompt, type LOGIN and press the RETURN key. When READY is displayed, the login operation has been ended successfully and the system enters the CLI mode.

If the user makes a login operation incorrectly, the system outputs the prompt ':::' again and the user must make a login operation again.
If no login operation is made within 30 seconds after the prompt ':::' is displayed, the following timeout error message is displayed and the user must start a login operation by pressing the RETURN key.

"Login time out error!!"

This timeout error message is displayed even if the user has typed only LOG within 30 seconds after the prompt ':::' was displayed.
When no free memory bank is available, the user cannot log into the system. To enter the CLI mode, at least one bank (64 KB) must be free.

"Sorry, main memory is not enough for you to login"

Terminal use states

Any terminal (including the main console) is in any one of the following states:

. Unused state
. Console use state
. File use state

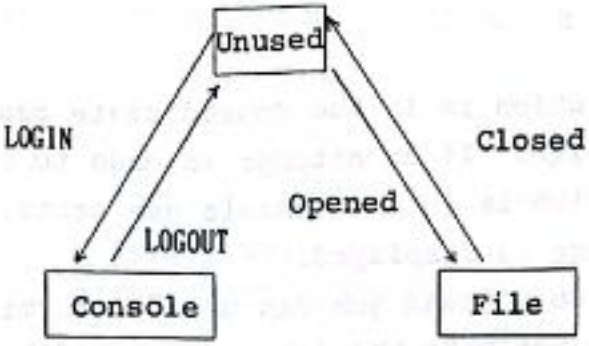


Fig. Terminal use states

Unused state

Any terminal is in the unused state immediately after the system power is turned on. The user cannot enter any CLI command from a terminal which is in the unused state.

Console use state

To enter CLI commands from a terminal, the terminal must be changed from the unused state to the console use state by making a login operation.
If a login operation is made on a terminal and the terminal is not used as a file, the system accepts the login request, gets the terminal into the console use state, displays READY, and waits for input of a CLI command.
If a login operation is made and the terminal is being used as a file, the system does not accept the login request and displays no error message.
To return the terminal from the console use state to the unused, a LOGOUT command must be entered. If the LOGOUT command is accepted, READY is not displayed.

File use state

A terminal which is in the unused state can be used as an I/O file. If an attempt is made to open a terminal which is in the console use state, an error message is displayed.
However, a foreground job can use the terminal which was used to activate the job as an I/O file even if the terminal is in the console use state.

Example:

```
READY
RASM ABC CRT/L/1
```

Turning the system power off

Check the following points before turning the system power off:

. All lgogging-in terminals display the READY prompt.

. Any I/O device such as a disk drive unit or printer does not operate.

Note that, if the system power is turned off during execution of a CLI command or a user program, MDOS/XMS may not be bootstrapped again.

CHAPTER 6 COMMAND LINE INTERPRETER (CLI)

6.1 Overview

The Command Line Interpreter (CLI) analyzes commands entered from keyboards.
CLI displays the READY prompt in any one of the following cases:

- After system initialization
- After system bootstrap
- After a keyboard break
- After execution of a program is completed

A CLI command line must be terminated by pressing the RETURN or ENTER key.
CLI consists of the following four units:

- Input unit
- Command file assembly unit
- Command file analysis unit
- Execution unit

CLI itself performs system call operations. That is, according to system calls, a command line is entered, and a command file is created, analyzed, and executed. When execution of the command is completed, the READY prompt is displayed and the system returns to the CLI mode.

6.2 Entry of More Than One CLI Command

The user can enter two or more CLI commands on one command line by inserting a semi-colon (;) between them as follows:

```
CLI command;CLI command;-----;CLI command
```

If more than one CLI command is entered on one command line, execution of the commands starts when the line is terminated by pressing the RETURN key. When execution of the first command is completed, CLI executes the next command immediately without expecting entry of a CLI command. This operation is repeated until the last command on the line and when execution of the last command, CLI waits for entry of another command line.
In this case, if an error occurs during execution of a CLI command other than the last, the remaining CLI commands are not executed, an error message is displayed, and the system returns to the CLI mode.
The user can enter up to 243 characters (including a RETURN code), that is, three lines and three characters on the screen (80 characters per line on the screen), for one command line. If an attempt is made to enter data exceeding the limit, that is, if a RETURN code is not entered at the 243rd character, CLI ignores data exceeding the 243rd character, displays only the first 243 characters, and outputs a warning sound.

During entry of CLI commands, the user can delete unnecessary data by one character by the DEL key or delete all CLI commands entered by the C key. When data is entered up to the last column of a line on the screen, the cursor automatically moves to the first column of thenext line and the user can continue the command line.

6.3 Indirect Specification of Command Lines

In a command line, the user can specify the name of a file to be executed as a command or argument by attaching @ at the head of the name.

Example:

Suppose that the following data is prepared,. using the editor, and it is written into a file with the name ASMBL:

```
AO.SR A1.SR A2.SR A/B SOUTA/L/1
```

To execute the file as a command, an assemble command must be entered as follows:

```
RASM @ASMBL
```

This command has the same meaning as the following command line:

```
RASM AO.SR A1.SR A2.SR A/B SOUTA/L/1
```

As shown above, it is useful to register long and frequently used command line as files (hereafter called command files).
Furthermore, by registering such command lines as command files, the user can actually enter data exceeding 243 characters which is the maximum number of characters which can be entered on one command line. This function allows any number of files to be specified so long as the number of characters on one command line (including semi-colons and a RETURN code) does not exceed 243. Any RETURN code in command files is ignored.

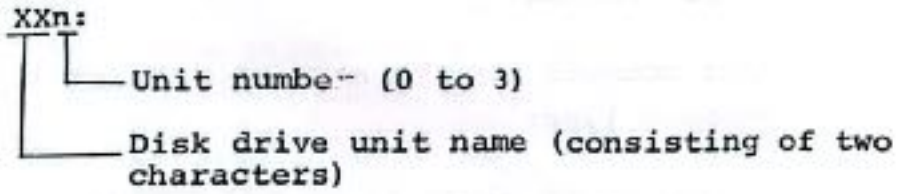
6.4 Specifying a Disk File

On a command line, a disk file can be specified in the following format:

Device number:[directory name(directory password)] filename(file password)

1 Device number

This specifies a disk drive unit.



A disk device name must be specified with one of the following:

FAN: Standard 1 MB floppy or 1 MB mini-floppy disk drive unit
WAn: 10 MB winchester fixed disk drive unit
WBn: 20 MB winchester fixed disk drive unit
WCn: 8 MB winchester fixed disk drive unit

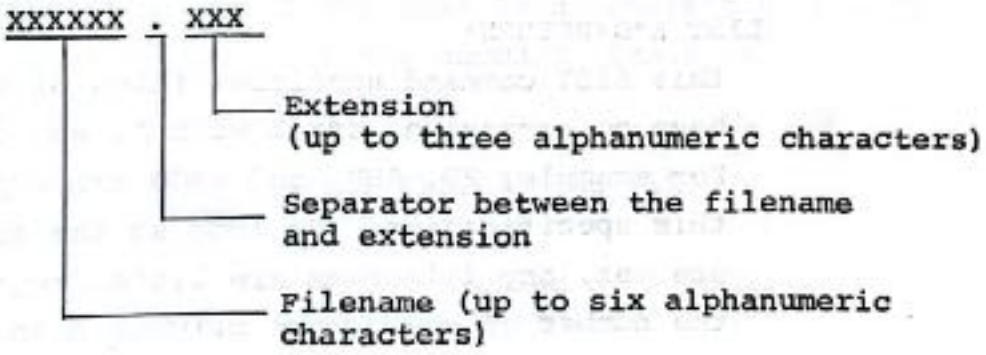
If no device number is specified, a disk drive unit with the disk containing the current default directory is automatically selected. If only a unit number is omitted like FA:, 0 is assumed.

2 [Directory]

This specifies the name of a file directory in which a file to be specified is registered. If a password is set for the directory, the password must be specified also. If no directory name is specified, the current default directory is assumed.

3 Filename

A filename is divided into a filename consisting of up to six characters and an extension consisting of up to three characters.



If a password is set for the file, the password must be specified, following the extension, with up to four alphanumeric characters.
Generally, a disk file can be specified in the format explained above in entry of a command line. But if a disk file is to be specified as a command argument. the user is allowed to use a wild card and a local current directory, although it depends on the CLI command to be used.

Wild card

A filename containing one or more question marks (?) and asterisks (*) is called a wild card.

? ..... Represents any one character.

* ..... Represents any one or more characters.

LIST A??D

This LIST command specifies files of which names have no extension and consist of four characters, starting with A and ending with D.

LIST ?.?<RETURN>

This LIST command specifies files of which names consist of only one character of filename and extension for each. As shown above, question marks (?) are used for reference of files of which the number of characters of names is known.

LIST A*D<RETURN>

This LIST command specifies files of which names have no extension, start with A, and end with D. For example, AD, ABD, and ABCD are included in this specification. So long as the above conditions are met, any filenames are listed regardless of the number of characters between A and D.

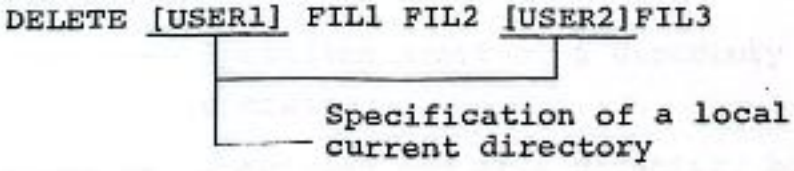
LIST A??*.*<RETURN>

This LIST command specifies files of which names start with A and consist of three or more characters. Question marks and asterisks can be used in one command like this.

For details on CLI commands which allow use of wild card and how to use them, see Chapter 7.

Local current directory

Generally, if a directory or disk drive unit name is omitted in specification of a file, the current default directory (including device names) is selected automatically. But, when a file is specified as a command argument, a local current directory replaces the current default directory temporarily within the command argument.
The local current directory can be set by entering only a device name and a directory name (or either of them) when specifying a file as a command argument. The device name and directory name (or either of them) compose a local current directory and if no directory name or disk drive unit name is specified after the local current directory is set, the contents of the directory are used as the default values. The local current directory remains valid until the next local current directory is specified or until the command itself is terminated.



The above example shows a local current directory specified in a DELETE command for deleting files.

If a directory name or disk drive unit name is omitted in a local current directory, the default value is taken from the current default directory. In the above example, if the device name in the current default directory is FA0: <SYS>, the contents of the local current directory are FA):[USER1], and FIL1 and FIL2 are assumed to be registered in the directory USER1. But FIL3 is specified with the directory USER2, so only the disk drive unit name in the local current directory is assumed and FIL3 is assumed to be registered in the directory FA0:[USR2]. Specification of a local command directory depends on the CLI command to be entered. For details, see Chapter 7.

CHAPTER 7 CLI COMMANDS

7.1 BDIR

The BDIR command is used to specify start or end of a directory back-up operation for each disk drive unit.

[Format]

BDIR/SW device number:SW device number:SW ---

. Global switch

Numeral --- Specifies the current default device.

S --------- Specifies start of a directory back-up operation.

E --------- Specifies end of a directory back-up operation.

D --------- Specifies display of information on directory back-up operations.

. Local switch

S --------- Specifies start of a directory back-up operation.

E --------- Specifies end of a directory back-up operation.

D --------- Specifies display of information on directory back-up operations.

None ------ Specifies display of information on directory back-up operations.

[Display]

Start and end times and dates (year, month, day, hour, minute, and second) of directory back-up operations, and a back-up operation currently being performed are displayed.

[Examples]

BDIR/S

This starts a directory back-up operation on the disk drive unit containing the current default directory.

BDIR/1 FA2:

This displays information on directory back-up operations on the disk drive units FA1 and FA2 which contain the current default directory.

7.2 CDIR

The CDIR command is used to register a user directory in the system directory SYS.DIR.

[Formats]

CDIR device number:[directory name(password)] maximum number of files which can be registered

CDIR device number:[directory name]maximum number of files which can be registered

- Global switch --- None
- Local switch --- None

Directory name

Specify a user-specified directory name. The extension can be omitted because the extension .DIR is automatically added.

Maximum number of files which can be registered

Any value can be specified as the maximum number of files which can be registered in the system directory so long as the directory space allows it. But a value greater than necessary should not be specified because more than one user uses the system directory. Otherwise, the system cannot be used efficiently.

The user should specify a multiple of 16 for this parameter. If the specified value is not a multiple of 16, MDOS/XMS adjusts it to a multiple of 16. For example, if 1 is specified, MDOS/XMS assumes 16.

Password

The user can specify a password for a user directory by using the CDIR command. If a password is once set for a user directory, the directory cannot be referenced without specifying the password. If a password specified for referencing the user directory is inconsistent with that set at registration, the user directory cannot be referenced. The password must consist of up to four alphanumeric characters.

If no password is specified for a user directory, the user directory can be referenced by any user because no password is set. The user should not specify a password for user directories which do not need security or protection.

[Examples]

CDIR FA1:[SATO]50

This registers the user directory SATO.DIR in the system directory (SYS.DIR) of the standard 1 MB floppy disk drive unit FA1:. It specifies 50 as the maximum number of files which can be registered, but it is internally set to 64 greater than 50 and nearest to 50 as a multiple of 16.

CDIR FA1:[SATO(ABC)]50

This registers the user directory SATO and sets the password ABC for the directory.

.3 CHATR

The CHATR command is used to change the file attribute, A, P, W, or R. However, onece the attribute A is set for a file, the attribute cannot be cancelled, that is, the file attribute cannot be changed.

[Format]

CHATR filename/SW

. Global switch ---- None
. Local switch ---- None

Each file attribute is represented by one alphabetic character as follows:

A: Attribute alteration disabled file (Attribute)

P: Permanent file (Permanent)

W: Write-protected file (Write protect)

R: Read-protected file (Read protect)

O: System file (Operating system)

I: Directory file (Directory)

S: Save file (Save)

C: Contiguous file (Contiguous)

The attribute P, W, or R is not set so long as it is not specified by the user.

[Example]

```
CHATR ABC.SR/P/R DATA.FIL/W
```

This sets the attributes P and R for the file ABC.SR and the attribute W for the file DATA.FIL.

7.4 COPY

Internally, when a COPY command is entered, one or more XFER commands are executed repeatedly to copy one or more files from one medium to another. That is, files are copied in units of a file. So when all files are copied to a medium, these files are squeezed in on the destination medium by removing any free space between them. Furthermore, the COPY command can perform the same processing as the MINIT and OSCHG commands at the same time by operation of the parameter switch. Therefore, the COPY command is useful when creating back-up files. The COPY command also allows files to be copied between different types of media. For example, it allows file transfer between a floppy disk and a winchester fixed disk. In data transfer from a large-capacity medium to a small-capacity one, data is copied until the small-capacity destination medium becomes full. When it is full, the following error message is displayed.

```
DISK SPACE EXHAUSTED.
```

[Formats]

```
COPY source medium destination medium

COPY/SW device number:
```

. Global switch
  C --- Specifies that the COPY program itself is to be copied.
  U --- Specifies that any system file such as MDOS/XMS, MINIT, or OSCHG is not to be copied when copying all files or the system directory.

. Local switch --- None

[Examples]

```
COPY FA0: FA0:
ILLEGAL CLI COMMAND
```

If the same device number is specified for the source and destination media, the above error message is displayed. The COPY command cannot be used for file transfer within the same device.

```
COPY FA0: FA1:
SET COPY DISKETTE AND STRIKE ANY-KEY
SYS  .DIR
KDOS .SYS ------ Copy completed.
OSCHG.SAV ------ Created, Copy completed.
MINIT.SAV ------ Created, Copy completed.
KSYSG.SAV ------ FILE DOES NOT EXIST
        LIST .SAV
             .
             .
             .
        BAS   DIR
             .
             .
             .
BAS  .DIR
       BASIC.LIB
             .
             .
             .
SET OS DISKETTE AND STRIKE ANY-KEY
READY
```

The above COPY command copies all files from FA0: to FA1:.
When a correct COPY command line is entered, the following message is displayed:

```
SET COPY DISKETTE AND STRIKE ANY KEY
```

In response to this prompt, the user needs to load a source medium and a destination medium to the specified devices respectively and to press a key.
As shown in the above example, the user can copy all files from one medium to another only by specifying device numbers in the format like FA0: FA1:.
When a COPY command specifying all files to be copied from one medium to another is entered, the destination medium is initialized (as done by the CLI MINIT command) and then the name of a directory to be copied is displayed. In this example, it is the system directory name SYS.DIR. When the source medium contains the MDOS/XMS operating system, it is also copied to the destination medium (as done by the CLI OSCHG command) and then the names of files to be copied are displayed one after another on the screen.
When the source medium contains a user directory file, the directory is copied and then files registered in the directory are copied. This is the case with the system directory. In this example, SYS.DIR and BAS.DIR show the case.
When all files are copied from the source medium to the destination medium, the following message is displayed:

```
  SET MDOS DISKETTE AND STRIKE ANY-KEY
```

When the destinationmedium is removed, the source medium is replaced with it, and a key is pressed, the system returns to the READY prompt.

```
  COPY/U FA0: FA1:
  SET COPY DISKETTE AND STRIKE ANY-KEY
  SYS     .DIR
            LIST  .SAV
                  .
                  .
                  .
            BAS   .DIR
                  .
                  .
                  .
  BAS     .DIR
            BASIC .LIB
                  .
                  .
                  .
  SET OS DISKETTE AND STRIKE ANY-KEY
READY
```

This command also specifies that all files are to be copied from FA0: to FA1:, but since the global switch U is specified, the COPY command does not perform the MINIT or OSCHG processing and only program files (such as LIST.SAVE and DISK.SAV) except system files are copied.

```
COPY WB:[SYS] FA:

SET COPY DISKETTE AND STRIKE ANY-KEY

SYS    .DIR
KDOS   .SYS ----- Copy completed.
OSCHG  .SAV ----- Created, Copy completed.
MINIT  .SAV ----- Created, Copy completed.
KSYSG  .SAV ----- FILE DOES NOT EXIST
          LIST   .SAV
                  .
                  .
                  .
   SET OS DISKETTE AND STRIKE ANY-KEY
READY
```

This command specifies that the system directory and files registered in the directory are to be copied. All files except directory files registered in the system directory are copied and when the source medium contains the operating system file, the file is copied first (as done by the CLI command OSCHG). (In this case, the destination medium is not initialized.)

```
   COPY/U WB:[SYS] FA:

   SET COPY DISKETTE AND STRIKE ANY-KEY

   SYS   .DIR
           LIST   .SAV
                   .
                   .
                   .
   SET OS DISKETTE AND STRIKE ANY-KEY
READY
```

In this example, the global switch U is specified unlike the above example. In this case, even if the source medium contains the operating system file, the file is not copied. All files except directory files registered in the system directory and the operating system file are copied.

```
COPY WB:[BAS] FA:
SET COPY DISKETTE AND STRIKE ANY-KEY
BAS  .DIR
       BASIC .LIB
          :
          :
SET OS DISKETTE AND STRIKE ANY-KEY
READY
```

This command specifies that files are to be copied in units of a directory. That is, all files registered in the directory BAS are copied in BAS.DIR of the destination medium. If there is no BAS.DIR directory on the destination medium, the BAS.DIR of the same size of the source medium BAS.DIR's is created on the destination medium. If the name of a file to be copied already exists on the destination medium, the file on the destination medium is overwritten even if the P attribute is set for the file.

```
COPY WB:[USER] MA:
SET COPY DISKETTE AND STRIKE ANY-KEY
USER  .DIR
        DATA1.#ORIGINAL DISKETTE UNMATCHED FILE
              PASS WORD
        DATA2.
        DATA3.
SET OS DISKETTE AND STRIKE ANY-KEY
READY
```

This command also specifies that files are to be copied in units of a directory, and in this case, a password set for a file is not specified correctly. When a password is set for a file on the source medium, the following message is displayed and the file is not copied:

```
#ORIGINAL DISKETTE UNMATCHED FILE PASS WORD
```

If an error occurs on the source medium as written above, the erroneous file is not copied and the system proceeds with transfer of the next file. If an error occurs on the destination medium, file transfer is terminates and the following message is displayed:

```
SET OS DISKETTE AND STRIKE ANY-KEY
```

```
COPY WB:[USER(PASS)](WORD) FA:
SET COPY DISKETTE AND STRIKE ANY-KEY
USER  .DIR
        DATA1.
        DATA2.
        DATA3.
SET OS DISKETTE AND STRIKE ANY-KEY
READY
```

This command also specifies that files are to be copied in units of a directory and a password is specified. To copy a directory or file with a password, the password must be specified as shown in this example. Therefore, if different paswords are set for files registered in the same directory, files must be copied in units of a file. The user can terminate file transfer by pressing the RESET or HALD key. When either of the keys is pressed during file transfer, the following message is displayed:

```
SET OS DISKETTE AND STRIKE ANY-KEY
COPY/C WB:
```

This command copies the COPY program itself to WB:. This copy function is available from the COPY program Rev.-00B.

Notes:

. Do not access any medium from or to which data is being copied.

. Every MDOS/XMS version does not allow execution of the COPY command. If a COPY command is executed under control of an MDOS/XMS version under which the COPY command cannot be executed, the following message is displayed and the system returns to the READY prompt:

OS VERSION ERROR

7.5 CREATE

The CREATE command is used to create a sequential or contiguous file from the keyboard.

[Format]

CREATE filename/SW file size

. Global switch ---- None

. Local switch

C ----- Contiguous file

None --- Sequential file

File size

The file size means the size of an area to be reserved on a disk when creating a contiguous file and is specified in either of the following format:

m*n or m

This specifies that a contiguous area of the size m*n bytes or m bytes is to be reserved. m indicates a record length and n indicates a number of records. The record length specified in this command is not always the same as that used when a program is accessing the contiguous file. As an example, to create a file of the record length 256 bytes and the number of records: 10, the user can enter 512*5 instead of 256*10. They output the same result. This is because the CREATE command for a contiguous file specifies the size of the file, not the record length. The record length used when a program is accessing a file is specified in the system call file open command ($OPEN). If the number of records (*n) is omitted, *1 is assumed. m and n must be within the following ranges respectively:

$1 \leqq m \leqq 65,535$

$1 \leqq n \leqq 16,777,215$

[Examples]

CREATE ABC

This creates the sequential file ABC. As shown in this example, for a sequential file, only the filename is registered and the number of blocks to be used need not be specified, that is, it is zero.

CREATE XYZ.DAT/C 15*300

This creates the contiguous file XYZ.DAT. A contiguous area of 4500 bytes (15*300) is created on the disk.

CREATE XYZ.DAT/C 4500

This CREATE command performs exactly the same processing as the above CREATE command.

CREATE CFIL/C 100 10,SFILE

This creates the contiguous file SFILF and sequential file SFILF.

7.6 CREV

The CREV command is used to set or change the creation date and time, and revision for a file.

[Format]

CREV filename date time revision

- Global switch --- None
- Local switch --- None

Date

The date must be specified in the order of day, month, and year by hyphening them year must be in the range from 0 to 99. If an incorrect value is specified for the month or day, for example, 13 is specified for the month or 30-2-83 is specified, an error message is displayed. The month can be specified with three upper-case English letters. For example, AFR can be used instead of 4.

Time

The time must be specified in the order of hour and minute by separating them by a colon (:).

Revision

The revision must be specified with up to three alphanumeric characters.

[Examples]

CREV ABC 7-7-83 1:00 00A

This changes the creation date of the file ABC to July 7, 1983, and the time to 1:00, and the revision to 00A.

CREV ABC 7-JUL-83 1:00 00A

This CREV command performs exactly the same processing as the above CREV command.

CREV ABC 7-7-83 1:00 00A,XYZ 5-5-82 2:00 00B

This CREV command performs the same processing as the above CREV command and, in addition, changes the creation date and time, and revision of the file XYZ. As shown in this example, when specifying two or more files in one CREV command, a comma (,) must be inserted between them.

CREV DEF 00C

This changes only the revision of the file DEF. When the creation date and time are omitted, the same date and time as before are assumed.

CREV GHI 9:30

This changes the creation time of the file GHI, the date remains the same as before, and the revision is deleted.

7.7 DATE

The DATE command is used to display or set the current date. The system has the calendar timer (year, month, day, hour, minute, and second), which can operate for about one week by the battery even if the computer power switch is turned off. If the current date need be set because the system has not been used for a long time, the DATE command must be used. The accuracy of the calendar timer is within ± one minute per month.

[Formats]

DATE ---------- For display

DATE DD-MM-YY ----- For setting

Date (day-month-year)

- Global switch ---- None
- Local switch ---- None

Date

The date must be specified in the order of day, month, and year by hyphening them. Year must be specified with thelower two digits, for example, if it is 1983, the year must be specified as 83. The year must be in the range from 0 to 99. The ranges of values which can be specified for the month and day are as follows:

Input ranges Year: 0 to 99
Month: 1 to 12
Day: 1 to 31

Abbreviations for months

1: JAN	4: APR	7: JUL	10: OCT
2: FEB	5: MAY	8: AUG	11: NOV
3: MAR	6: JUN	9: SEP	12: DEC

The maximum value which can be specified for the day depends on the month. For example, 30 can be specified for Janurary, but it cannot be specified for February. If 30 is specified for February, an error message is displayed. As shown above, the month can be specified with three upper-case English letters instead of a number. For example, APR can be specified instead of 4. The month is always displayed with an abbreviation of three upper-case English letters.

[Example]

DATE

This displays the current date.

DATE 7-JUL-83

This sets the current date to July 7, 1983.

DATE 7-7-83

This DATE command has exactly the same processing as the above DATE command.

Notes:

. Even if 0 is specified for the year, month, and day, no error message is displayed.

. If both the current time and date are to be set, they must be specified in the order of time and date. If they are specified in the reverse order, the time will change the specified date.

7.8 DDIR

The DDIR command is used to delete a user directory and all files registered in the directory at the same time. When deleting them, this command ignores any file attribute even if it is P.

[Format]

DDIR device name:[directory name(password)]

- Global switch ---- None
- Local switch ----- None

Directory name

The directory name must be specified. If no directory name is specified or if the current default directory name is specified, an error message is displayed. The system directory (SYS.DIR) cannot be deleted.

[Examples]

DDIR [DIRCT]

This deletes the directory DIRCT from the current default device. If a password is set for the directory, an error message is displayed because no password is specified. That is, if a password is set for a directory, the password must also be specified.

DDIR FA0:[DIRCT.SAV(KEY)]

This deletes the directory DIRCT.DIS (password: KEY) from the floppy disk No.0. The extension .SAV of the directory is automatically replaced by .DIR.

7.9 DELETE

The DELETE command is used to delete one or more files from the specified directory. It can also be used to delete M200 series mark-V and mark 41 files. If a password is set for a file to be deleted, the password must also be specified.
If a file registered a directory other than the current default directory is to be deleted, the directory name or a local current directory must be specified.

[Format]

DELETE/SW device name:[LDIR]/SW FILE1/SW FILE2/SW

LDIR ---- Name of a directory in which the file to be deleted is registered

FILE ---- Name of the file to be deleted

. Global switch

P --- Specifies that the file is to be deleted even if it has the P attribute (permanent file).

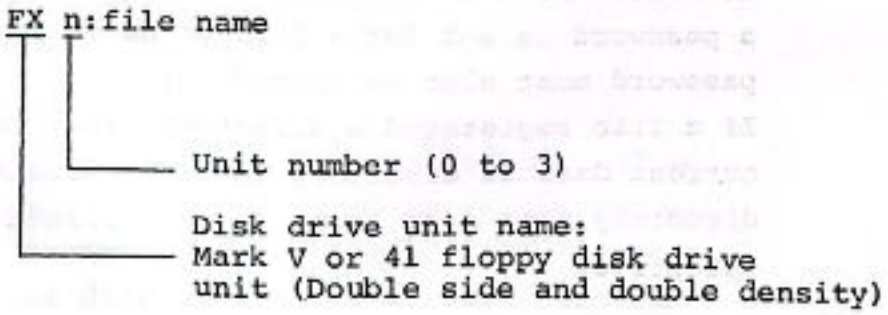
C --- Enables the user to specify whether files included in the argument are to be deleted or not individually. This specification can be made by pressing the D key. That is, when the D key is pressed after a filename is displayed, the file is deleted.

. Local switch

C --- Enables the user to specify whether the file or wild card for which this switch is set is to be deleted or not.

Filename

An M200 series file is specified in the following format:



A wild card can be used for M243EX and M200 series files.

[Examples]

(1) DELETE ABC.SAV DEF FA1:[USER(PASS)]
GHI.SR(WORD)

```
FA0:[SYS   .DIR]ABC    .SAV ---- Deleted
FA0:[SYS   .DIR]DEF         ---- FILE DOES NOT
                                 EXIST
FA1:[USER .DIR]GHI     .SR  ---- Deleted
```

This deletes the files ABC.SAV and DEF from The current default directory FA0:[SYS.DIR], and the file GHI.SR (password:WORD) from the directory FA1:[USER.DIR](password:PASS). This example shows that the file DEF is not registered in the directory FA0:[SYS.DIR]. As shown in this example, if no local current directory is specified, the current default directory is assumed.

(2) DELETE FA1:[USER] A*.* DEF.SR [SYS]ABC.SAV(123)

```
FA1:[USER   .DIR]ABC1 .SR   ---- Deleted
FA1:[USER   .DIR]A111 .SAV  ---- Deleted
FA1:[USER   .DIR]DEF  .SR   ---- Deleted
FA1:[SYS    .DIR]ABC  .SAV  ---- UNMATCHED FILE
                                 PASS WORD
```

In this example, a local current directory and a wild card are specified. A local current directory consists of only a device name and a directory name, or either of them, and does not contain a file name. In this example, FA1:[USER] is a local current directory. For the filenames following the local current directory, FA1:[USER] is assumed as the device and directory names.
The wild card A*.* indicates files which are registered in the directory FA1:[USER] and do not have the P attribute, and of which names start with A. Therefore, for example, if the file APU.SAV with the P attribute is registered in the directory, the file is not deleted and its name is not displayed too.
To delete the file APU.SAV, the global switch /P must be set.
The file ABC.SAV is not deleted because the password set for the file is not specified.

(3) DELETE/P FX1: *.SR/C *.WK

This DELETE command specifies that M200 series files are to be deleted. When this command is executed, the names of files (including those with the P attribute) which are covered by the wild card *.SR and stored on the M200 series medium in the mark V (or 41) unit 1 are displayed one by one and the user is requested to specify whether each file is to be deleted or not. To delete the file, the user must pressed the D key. If the file is not to be deleted, the user need to press a key other than the D key so as to proceed to the next file.

When processing of all files covered by *.SR is completed, files covered by the next wild card *.WK are deleted.

7.10 DIRT

The DIRT command is used to specify the current default directory.

[Formats]

DIRT device number:[directory name(password)]

--- For specification of the current default directory

DIRT --- For display of the current default directory name

. Global switch --- None
. Local switch --- None

The current default directory implicitly links the system to its user. That is, the user can reference any file registered in the current default directory without specifying the directory name. It can be set for each user terminal.
When frequently referencing files in a specific directory, the user should specify the directory as the current default directory by the DIRT command. Once this is done, the user can reference any file in the directory only by specifying the file name.

[Examples]

DIRT FA0:

This specifies the system directory (SYS.DIR) on the standard 8 inch floppy disk in Unit 0 as the current default directory.

DIRT WA0:[SATO(PASS)]

This specifies the directory SATO (password:PASS) on the 10 MB winchester fixed disk as the current default directory.

7.11 DISK

This DISK command is used to display the numbers of used and unused blocks for one or more disks. One block is 512 byte long.

[Format]

DISK device number: device number: device number:

- Global switch --- None
- Local switch --- None

[Display format]

The numbers of used and unused blocks are displayed as follows:

LEFT:XXXXXX USED:XXXXXXXX

Total number of blocks

The total number of blocks (used and unused) depends on the disk type.

[Examples]

DISK

This displays the numbers of used and unused blocks reserved for the current default directory.

DISK FA1:

This displays the numbers of used and unused blocks of the disk in the device number FA1:.

7.12 DUMP

The DUMP command is used to dump the contents of one or more files in units of a block and in hexadecimal or character codes.

[Format]

DUMP filename output filename/L

. Global switch

A or J ---- Specifies that the file is to be dumped in character codes.

A: ASCII '20' to '7E'
J: JIS '20' to '7E' and 'A0' to 'DF'

If data to be dumped is not represented by any of the above codes, it is dumped in hexadecimal codes.

None ------ Specifies that the file is to be dumped in hexadecimal codes.

. Local switch

L ------ Specifies that the file is an output file.

Number-- Specifies a mode if data is to be dumped to an I/O file.

A or J-- Specifies that the file is an input file and is to be dumped in character codes regardless of the global switch.

None --- Specifies that the file is an input file and is to be dumped in codes specified by the global switch.

Output file

The user can specify either a disk file or an I/O file as an output file. If no output file is specified, the contents of the file are displayed on the screen.

Output control

If no output file is specified, the first 22 lines of data are displayed on the screen and data output stops. To continue data display, a key other than the END key must be pressed and the number of lines to be displayed depends on which key is pressed as follows:

. When the H key is pressed, the next 11 lines are displayed.

. When the LF key is pressed, the next one line is displayed.

. When a key other than the H and LF keys is pressed, the next 22 lines are displayed.

If the END key is pressed at this point, the system terminates execution of the DUMP command and returns to the READY prompt.

Block length

One disk file block is 512 byte long, but for a sequential file, 504 bytes excluding link bytes are handled as one block.

[Example]

DUMP FILE1/J FILE2,FILE3 POUT/L/1

FILE1 --- Is displayed on the screen in character codes (JIS).

FILE2 --- Is displayed on the screen in hexadecimal codes.

FILE3 --- Is dumped to the output file POUT in hexadecimal codes.

7.13 ERMSG

The ERMSG command is used to specify in which language, English or Japanese, error messages are to be displayed.

[Format]

ERMSG/SW

. Global switch

E --- Specifies English.
K --- Specifies Japanese (katakana).

. Local switch --- None

[Examples]

ERMSG/E

This specifies that error messages are to be displayed in English.

ERMSG/K

This specifies that error messages are to be displayed in Japanese.

14 FILE

The FILE command is used to display the directory status, that is, the total number of files which can be registered, the number of files registered (used), and the number of spaces left for file registration (left) for one or more specified directories.

[Format]

```
FILE device:[directory name(Password)] .....
output filename/L
```

[Switches]

. Global switch --- None

. Local switch

L ---------- Specifies that the file is an output file.

Number ----- Specifies a mode if an I/O file is to be used as an output file.

None ------- Specifies that the name is a directory name of which filename registration status is to be displayed.

[Display format]

"FA0:[USER .DIR] TOTAL 0032,Used 0010,LEFT 0022"

This means that the directory USER in the device number FA0: allows up to 32 files to be registered in it, 10 files are currently registered, and 22 more files can be registered.

[Examples]

1) FILE

This displays the file registration status of the current default directory on the screen. If no output file is specified as shown in this example, data is displayed on the screen and if no directory is specified, the current default directory is assumed.

2) FILE FA1:[SYS] WB0:[USER] WB0:[SYS] POUT/L/1

This outputs the file registration status of the directories FA1:[SYS], WB0:[USER], and WB0:[SYS]. As shown in this example, two or more directory names can be specified by inserting a space between them.

Notes:

. If an incorrect output filename is specified or if a directory is specified in a wrong format, the following message is displayed:

ILLEGAL DATA ON COMMAND LINE

Error messages which may be output during execution of the FILE command are covered by MDOS/XMS error messages.

. If a password is specified for a directory to be specified, the password must be specified as well.

FUNC

The FUNC command is used to specify and change characters for functions keys of the main console or terminal keyboard, and display and print the function key table.

[Formats]

(1) FUNC function key filename

If a FUN command is entered in this format, the function key file is set in MDOS/XMS and the system returns to the READY prompt.

(2) FUNC

If a FUNC command is entered in this format, the #prompt is displayed and the user needs to enter any one of the following commands in response to the prompt:

[Commands]

1) L (List)

This command displays the function key table, enclosing it by colons. If data in the table is represented by codes other than those of the G0 and G1 character sets, it is displayed in hexadecimal like <'0D>. If the number of characters registered for each key is 33 or less, the contents are displayed on the screen, separating them into two columns: Fn (not shifted) and fn (shifted). Data output stops when one screen of data is displayed. The next processing to be done depends on which key is pressed as follows:

. When the LF key is pressed, the screen is scrolled up by one line.

. When the END key is pressed, theuser exits from the L command.

. When a key other than the LF and END keys is pressed, the next one screen of data is displayed.

2) Fn (Function key registration)

3) fn (function key registration)

In Items 2) and 3), n indicates a function key number, which must be in the range from 00 to 18. Up to 73 characters can be specified for each function key. (A control code is not counted as one character.)

4) CFn (Extended function key registration)

System command keys above the ten keys can be used as function keys by setting the system to the extension function mode. The default value is OFF (invalid).

5) R (Read)

This reads the function key file into the table. If the extension is omitted in specification of the function key file, the default extension .FNC is assumed.

6) W (Write)

This writes the function key file on to a disk under a specified name. The default extension is .FNC.

7) P (Print)

This transfers the function key file to an output file. If an I/O file is to be used as an output file, the system mode must be set to SWITCH.

8) E (End)

This loads the function key table into MDOS/XMS and terminates execution of the FUNC command.

9) e (end)

This terminates execution of the FUNC command without loading the function key table into MDOS/XMS.

[Examples]

1) FUNC

```
#List

 ********************************************

F00                         f00
F01                         f01
F02                         f02
 :                           :
 :                           :
F15                         f15
F16                         f16
F17                         f17
F18                         f18
F19                         f19

[F1]
 :
 :
[F4]
 ********************************************
```

Explanation

F1 to F15 and f1 to f15:
Used by the function keys F1 to F15 of the RT-20 keyboard (SORD).

F16 to F18 and f16 to f18:
Used by the RUN, HALT, and CONT keys of the RT-20 keyboard when the system is in EFM.

F16 to F19 and f16 to f19:
Used by the function keys F1 to F4 of the VT-100 (DEC).

[F1] to [F4]:
Used by the RUN, HALT, CONT, and RESET keys of the M243EX when the system is in EFM.

2) #F01: BASIC

#F02: BASIC

These register BASIC for F1 and f2.

Note: Each command must be terminated by pressing the RETURN, ENTER, or END key. When the RETURN or ENTER key is pressed, the CR code ('0D) is also registered. When the END key is pressed, no CR code ('0D) is registered.

3) F10:<27>]7m

This registers ESC]7m for the F10 key. As shown in this example, a control code must be enclosed by < >. A decimal number can be entered as it is, bu a hexadecimal number must be entered with ' as shown below:

Examples	<10>	Decimal
	<'10>	Hexadecimal

4) Print-out file name:OUT/I

This outputs the contents of the function key file to the output file POUT (printer).

Note: The system is set to and released from the Extension Function Mode (EFM) according to the following escape sequences:

Set ---------- ESC]=8h
Release ------ ESC]=8l

7.16 LIST

The LIST command is used to output registered filenames in a list format for one or more directories at the same time. This command is valid also for M200 series media (Mark V and 41 floppy disks) and allows use of local current directories and wild cards.

When a LIST command specifying that data is to be displayed on the screen is executed, the first one screen (22 lines) of data is displayed and then data output stops. To continue data output, a key other than the END key must be pressed. In this case, the number of lines to be displayed depends on which key is pressed as follows:

- When the H key is pressed, the next 11 lines are displayed.
- When the LF key is pressed, the next one line is displayed.
- When a key other than the H, LF, and END keys is pressed, the next 22 lines are displayed.

To terminate execution of the command and return to the READY prompt, the user must press the END key.

[Format]

LIST/SW device number:[directory name(password)] wild card output file/L

[Switches]

1) Global switches

A -------- Specifies that the names of files with the P attribute are also to be output.

S -------- Specifies that filenames are to be stored and output in the order of internal codes.

2) Local switch

L Specifies that the file is an output file.

Number --- Specifies a mode if an I/O file is specified as an output file.

None ----- Specifies that the entry item is a wild card.

[Device number]

M200 series media are specified in the following format:

FXn:

n Unit number (0 to 3)

FX Disk drive unit name:

8 inch floppy disk (double side and double density)
(Mark V and 41 only)

[Examples]

1) LIST

This displays a list of the names of files registered in the current default directory and which have no P attribute.

2) LIST/A/S [USER(ABC)] *.SAV SOUTA/L/1

This sorts filenames registered in the directory USER.DIR (password: ABC) and covered by the wild card *.SAV (save files) and outputs them to the file SOUTA.

3) LIST/A FX1: A*.*

This outputs filenames registered on either the M200 mark V or 41 floppy disk in Unit 1 and which start with A. The names of files with the P attribute are also output. For M200 series media, no directory name is to be specified because they have no user directory.

4) LIST WB0:[USER] AB*.SR AB*.RB AB*.SAV

This outputs the names of files registered in the directory WB0:[USER], and which have no P attribute and are covered by the wild cards AB*.SR, AB*.RB, and AB*.SAV, sequentially.

Note: Take care not to confuse FX for M200 with FA for M24 when specifying a device name. If it is incorrectly specified, the error message 'DISK FAIL' is output.

7.17 MINIT

[Functions]

The MINIT command is used to initialize a medium which contains operating system files. At the same time, all program data files are deleted from the initialized medium, which now can be used as a data medium.

This command is also used when creating back-up operating system files or updating operating system files. This can be done by executing the OSCHG command following the MINIT command.

In addition, the user can double the directory area, that is, the number of filenames which can be registered in the directory, by specifying the global switch E in this command. But this function should not be used together with the directory back-up function (CLI BDIR command). If a BDIR command is executed for a medium on which the directory area is doubled, the command operates but no processing is performed on the medium.

<Precaution>

Note the following points when using M243EX floppy disks: Do not call any program from a floppy disk which has the double directory area under control of MDOS/XMS bootstrapped from another disk which has the usual directory area and for which the directory back-up function is specified. If the operation is done, the directory back-up function operates and the doubled directory area is destroyed. To execute a program on the disk which has the double directory area, MDOS/XMS must be bootstrapped again from the disk and the program must be called.

[Formats]

MINIT/SW
MINIT device number:

[Switches]

- Global switches

  Number --- Specifies a drive number of the current default device.

  E -------- Specifies that the directory area is to be extended.

- Local switch

  None

[Examples]

(1) MINIT/1

This initializes the medium in the current default device Unit 1.

(2) MINIT ZA1:

This initializes the medium AZ1:

(3) MINIT/E FA1:

This initializes the medium FA1: and doubles the directory area, thus enabling registration of 480 filenames.

Notes:

(1) This CLI command must not be executed while another job is being executed, that is, it must be processed singly. If this command is executed while another job is being executed, it will affect system operation and cause disks to disks being operated.

(2) This CLI command just logically initializes a medium and does not physically initialize it. That is, it cannot format it. So it must be executed for a formatted disk.

(3) The MINIT command cannot initialize a medium which contains the current default directory or the bootstrap program (system default medium). Therefore, it must not be executed on a system which has only one disk drive unit.

(4) Some MDOS/XMS versions do not allow execution of the MINIT command. If it is entered under control of such a MDOS/XMS, the following message is displayed and the system returns to the READY prompt:

OS VERSION ERROR

(5) MDOS/XMS cannot be bootstrapped from an initialized program.

7.18 OSCHG

The OSCHG command is used to copy the bootstrap loader or operating system from the system default medium to another medium. This command is used when updating operating system files or creating back-up operating system files.

It does not transfer any CLI command file such as LIST or DISK, or any system program file such as BASIC. If it is required, the XFER command must be used.

Basically, the OSCHG command is for updating operating system files. The user can create a back-up medium of the original MDOS/XMS medium by executing this command following the MINIT command.

[Formats]

OSCHG/SW
OSCHG device number:

- Global switch
  Number --- Specifies a drive number of the current default device. The user does not need to specify a device number.
- Local switch --- None

[Examples]

OSCHG/1

This copies the bootstrap loader and operating system from the system default medium to the current default device Unit 1.

OSCHG WA:

This copies the bootstrap loader and operating system from the system default medium to WA:.

Notes:

- The OSCHG command must not be executed while another job is being executed, that is, it must be executed singly. If it is executed while another job is being executed, it will affect system operation and cause errors to disks being operated.
- A medium from which operating system files are to be copied must contain them, of course. A medium to which these files to be copied must be initialized by the MINIT command beforehand and must not contain any file.
- Operating system files cannot be copied to a medium which contains the current default directory or the bootstrap program (system default medium). Therefore, it must not be executed on a system which has only one disk drive unit.
- The MINIT or OSCHG program is not copied if the system default medium does not contain them or if a medium to which they are to be copied already contains the same version of the MINIT and OSCHG program.
- When operating system files are copied to a medium by this OSCHG command, any system program or user file stored on the files is not affected.
- Some MDOS/XMS version do not allow execution of the OSCHG command. If it is entered under control of such a MDOS/XMS version, the following message is displayed and the system returns to the READY prompt:

  OS VERSION ERROR

7.19 PASS

The PASS command is used to set or change a password for a user directory or file. To change a password, the old password must be specified correctly and the file must not be protected.

[Formats]

PASS directory name new password/D
PASS filename new password/F
PASS filename new password/D new password/F

- Global switch --- None
- Local switch
  D --- Specifies that the entry item is a new directory password.
  F --- Specifies that the entry item is a new file password.

[Examples]

PASS [USER(PAS1)] PAS2/D

This changes the password of the user directory USER.DIR from PAS1 to PAS2.

PASS FILE LOCK/F

This sets the password LOCK for the user file FILE.

PASS [USER(PAS1)] FILE PAS2/D LOCK/F

This changes the password of the user directory USER.DIR from PAS1 to PAS2 and sets the password LOCK for the file FILE registered in the directory.

PASS [USER(PAS1)] FILE(LOCK) /D/F

This cancels the password PAS1 of the user directory USER.DIR and the password LOCK of the file FILE.

7.20 POFF

The POFF command is used to prevent data from being written on to a winchester fixed disk such as (WA or WB) when turning the system power off. So this command must be entered before the system power is turned off.

When it is executed, the read/write head in the winchester fixed disk drive unit moves to the most inner cylinder position which is not used by the system. As a result, even if the read/write head writes data into the cylinder by mistake when turning the system power off, any program data is not affected because it is outside the system area.

When this command is entered, the system displays the following message before the read/write head moves:

```
CHECK NO OTHER JOBS ! OK (Y/N) ?
```

To continue execution of the command, check that no job is being executed and type Y in response to this message. If the N or RESET key is pressed, the head does not move and the system returns to the READY prompt. When Y is entered and the read/write head moves to the appropriate position on the winchester fixed disk, the following message is displayed and the system stops:

```
SYSTEM POWER OFF
```

[Format]

```
POFF
```

7.21 RENAME

The RENAME command is used to change one or more filenames at the same time. The names of files with the P attribute cannot be changed. Any password cannot be changed either.

[Format]

RENAME old f-lename new filename

- Global switch --- None
- Local switch ---- None

[Examples]

RENAME ABC.SAV XYZ.SAV

This changes the name of a file in the current default directory from ABC.SAV to XYZ.SAV.

RENAME FA1:[DIRECT]ABC.SAV(KEY) XYZ.SAV

This changes the name of a file (password: KEY) in the directory DIRECT on the device number FA1: disk from ABC.SAV to XYZ.SAV. The password KEY is not changed.

RENAME 1:ABC AAA,XYZ.SAV ABC.SAV

This changes the name of a file in the current default directory in the current default device Drive 1 from ABC to AAA, and the name of another file in the current default directory in Drive 0 from XYZ.SAV to ABC.SAV.

Note: Any device number must be specified immediately before an old filename. If it is specified immediately before a new filename, it is ignored.

7.22 REZERO

The REZERO command is used to moves the read/write head to the Cylinder 0 position. When moving the system, the read/write head in the winchester fixed disk drive unit must be locked. To lock it, the head must be placed at the Cylinder 0 position. Therefore, the user should execute a REZERO command and then turn the system power off to lock the read/write head correctly.

[Formats]

REZERO

REZERO/SW device number:

- Global switch
  Number --- Specifies a drive unit of the current default device.
- Local switch --- None

[Examples]

REZERO

This moves the read/write head to the Cylinder 0 position in the current default device.

REZERO/1 WA0: WB0:

This moves the read/write heads to the Cylinder 0 position in the current default device Drive 1, and WA0: and WB0:.

7.23 SGL

The standard M243EX mark series system provides the graphic function, which the user can use easily by the SORD Graphic Language (SGL).

To use SGL, the user must load the SGL program to memory by the SGL command, because SGL is not loaded at the same time as MDOS/XMS is bootstrapped.

Therefore, the user needs to reserve at least one memory bank at system generation. SGL operates as a memory file.

[Formats]

SGL

This loads the SGL program to memory.

SGL OFF

This deletes the SGL program from memory.

SGL device number:

This transfers the SGL program to a specified device.

[Examples]

SGL

This writes the SGL program into the memory file SGL.SYS, thus enabling execution of SGL graphic commands.

SGL OFF

This deletes the memory file SGL.SYS, thus disabling execution of SGL graphic commands.

SGL FA1:

This transfers the SGL programs, SGL.SAV and SGL.SYS, to the floppy disk drive unit 1. But if the disk in Unit 1 already contains the same or later version of the SGL programs, they are not transferred.

7.24 SIO

The SIO command is used to display or alter the data transfer rate, mode, parity check, the number of stop bits, and the numbers of transmission and receipt data bits at one or more serial ports. A serial port name is specified with any one of A, B, C, and D.

[Formats]

SIO port name port name --- For display
SIO port name/alteration item/alteration item/
--- For alteration

- Global switch ---- None
- Local switches --- Used only for alteration.

Alteration items

Alteration items are represented by B (data transfer rate), M (mode), P (parity check), S (the number of stop bits), T (the number of transmission data bits), and R (the number of receipt data bits). Items which are specified as alteration items are only changed.

- B (Data transfer rate)

  The data transfer rate can be specified with one of the 16 baud rates listed below (50 to 19,200 baud). One serial port consist of an input port and an output port, but the data transfer rate cannot be set for each of them.

  50
  75
  110
  150
  200

300
600
1200
2400
4800
9600 ..... default value
19200

- M (Mode)

  Specify either W (WAIT mode) or I (IGNORE mode).

  WAIT mode:
  In the WAIT mode, data is sent only when the input port of the CN-4 (Port A, C, or D) or CN-5 is on. If the input port of the CN-4 or CN-5 turns on after the first stop bit in a character being sent is detected, character transmission stops after the next character is sent. The WAIT mode is used when a printer stops data transmission while it is receiving data.

  IGNORE mode:
  In the IGNORE mode, data is sent regardless of the CN-4 or CN-5 input port state.

- P (Parity check)

  Specify O (odd parity), E (even parity), or N (no parity).

- S (The number of stop bits)

  Specify 1, 1.5, or 2. The default value is 2.

- T (The number of transmission data bits)

  Specify 5, 6, 7, or 8. The default value is 8.

- R (The number of receipt data bits)

  Specify 5, 6, 7, or 8. The default value is 8.

[Examples]

SIO A B

This displays the current data transfer rates, modes, parity check methods, the numbers of stop bits, the numbers of transmission data bits, and the numbers of receipt data bits at the serial ports A and B.

SIO A/B=9600/M=W/P=N/S=2/T=8/R=8

This sets alteration items at the serial port A as follows:

```
Data transfer rate:  9600 baud
Mode:  WAIT mode
Parity check:  None
Number of stop bits:  2
Number of transmission data bits:  8
Number of receipt data bits:  8
```

Note: The SIO command cannot be used when a serial port is used as a terminal or when it is open as an I/O device.

7.25 SLICE

In multijob processing, after executing a job for a fixed time period (time slice), MDOS/XMS stops the job temporarily and proceeds to execution of another job.

The SLICE command is used to change the time slice. By changing the time slice, the total processing efficiency may be improved, depending on the job.

[Formats]

SLICE ----- For display
SLICE n --- For setting

- Global switch --- None
- Local switch ---- None

The time slice assigned to each job is 10 ms multiplied by n. n must be in the range from 0 to 255. If 0 is specified, multijob processing is not done according to the time slice. The default value is 10 (100 ms).

[Examples]

SLICE

This displays the current time slice.

SLICE 5

This sets the time slice to 50 ms.

7.26 TIME

The TIME command is used to display or set the time. The system has the calender timer (year, month, day, hour, minute, and second) and even if the computer power switch is turned off, the timer can operate for about one week by the battery. Therefore, if the system has not been used for a long time over one week, the calender timer should be set. The accuracy of the timer is within ± one minute per month.

[Formats]

TIME ---------- For display
TIME/SW HH:MM --- For setting

where HH represents hour and MM represents minute.

- Global switch

  A --- Specifies that the calender timer is to be set by the key input. If A is not specified, it is set back about one second for loading and executing the TIME program itself after entry.

- Local switch --- None

The time must be specified in the order of hour and minute and by separating them by a colon (:). One day is counted as 24 hours. Hour and minute must be in the following ranges respectively:

Hour: 0 to 23
Minute: 0 to 59

The unit of seconds is always set to zero.

[Examples]

TIME 8:4

This sets the timer to 8:04 a.m.

TIME 08:04

This sets the timer to 8:04 a.m.

TIME 16:0

This sets the timer to 4:00 p.m.

TIME/A 12:00

This sets the timer to the noon.

If the SPACE key is pressed when the following message is displayed, the timer is set to the specified time:

Please touch space.

7.27 VOL

Every floppy disk and winchester fixed disk supported for the M24 are provided with a volume block consisting of four areas. The VOL command is used to read data from or write new data into the volume block.

The four areas of the volume block contains the following information respectively:

1) User identification name (User volume Name)

   This area is also called the volume label and contains a disk user name.

2) Creation date (Data Volume Created)

   This area contains a date when use of the disk started.

3) Use purpose (Volume Purpose)

   This area contains data on the use purpose and application of the disk and is also called the header label.

4) Volume password (User Volume Password)

   A user password is set in this area for preventing the contents of the disk from being copied by unauthorized users.

The characters enclosed by parentheses ( ) are used as the area name for display.

[Formats]

1) VOL device number: output file name/L --- For reading

2) VOL/W device number: ---------- For writing

[Switches]

1) Global switches

W ------ Specifies that new data is to be written into the volume block.

None --- Specifies that data is to be read from the volume block.

2) Local switch

L ------ Specifies the output file into which data is to be read from the volume block.

Number - Specifies a mode if an I/O file is specified as an output file.

[Examples]

1) VOL

This reads the volume block on the disk registered in the current default directory and displays its contents on the screen.

2) VOL WB0: POUT/L/1

This outputs the contents of the volume block on the 20 MB winchester fixed disk WB0: to the output file POUT.

As an example, data is displayed on the screen or output to an output file in the following format:

```
  *** Volume Block of FA1: ***

User Volume Name     :John Smith: --->  The user name is out-
                                        put between colons.
Date Volume Created  :07-SEP-82:
Volume Purpose       :MYFRIEND:
User Volume Password::             --->  No volume password
                                         is set.
```

3) VOL/W FA1:

This specifies that new data is to be written into the volume block on the disk FA1:. When this command is entered, the current contents of the block are displayed and then the system waits for data entry on the next line. If data enclosed by parentheses ( ) is entered or if the CR or END key is pressed, execution of the command is terminated. Pressing the CR key leaves the previous contents as they are. Pressing the END key deletes the contents. If the T key is only pressed when entering a date, the date of the day when the command was executed is entered.

When specification of all items is completed, the following message is displayed:

Volume Password?

If the Y key is pressed in response to this message, the newly specified data is written into the volume block on the specified disk and the system returns to the READY prompt. If the END key is pressed, the system returns to the READY prompt immediately without writing the data. If the N key is pressed, the user needs to enter new data again from the first item.

4) VOL WC0:

Protected by User Volume Password!

This specifies that the contents of the volume block on the mark X disk are to be displayed on the screen, but since a volume password is set for the block, the above message is displayed. Following the message, the system displayes the prompt:

Volume Password?

In response to this prompt, enter the password. When it is entered correctly, the system proceeds to execution of the command. But if it is entered incorrectly, the system displays the above prompt again. If the passwofd is entered incorrectly three times, the system displays the following message and returns to the READY prompt:

Unmatched Volume Password!

Note: If an attempt is made to execute the VOL command for a disk other than the M24 series mark V and 41, the system displays the following message and returns to the READY prompt:

Device number: Illegal Disk!

7.28 VCOPY

The VCOPY command is used to copy all images from one medium to another, but it must be used between disks of the same type unlike the COPY command. This command is valid only for Mark-V, Mark-41, and Mark-IV disks of the M24 and M243 series. It copies data in units of a cylinder, not in units of a file, so any space between files is not removed in data transfer and files are not squeezed in on the destination medium. (The SWEEP processing is not performed.)

The VCOPY command copies the same amount of data faster than the COPY command. The following table shows differences between the VCOPY and COPY commands and the user should use them appropriately according to the purpose:

Program name	VCOPY SAV	COPY SAV
	Mark-V 2.5 minutes to 10 minutes Mark-IV 3.5 minutes to 8 minutes	Mark-V 10 minutes to 28 minutes Mark-IV 20 minutes to 45 minutes
Required data transfer time	For the VCOPY command, the required data transfer time does not depend on the number of files on the medium, but it depends on the sector sequence. The VCOPY command can copy data in the shortest time between media both of which sector sequences are 1.	
Internal processing	Data is read and written in units of a cylinder. • All images must be copied. • Files are not squeezed in. (The SWEEP processing is not done.) • Any file password is ignored. • The volume password must be specified.	A MINIT command and an OSCHG command are executed, and then XFER commands are executed. • Data can be copied in units of a directory. • Files are squeezed in. (The SWEEP processing is done.) • All file passwords must be specified. • The volume password is ignored.
Disk drive units which can be used	Between disks of the same type, for example, between FA: and FA:	Between disks of different bypes such as FA:, WA:, WB, and WC:

[Format]

VCOPY source disk drive unit number: destination disk drive unit number

[Operation procedure]

(1) Enter a VCOPY command in the format specified above. If different disk types are specified or if the same unit number is specified, an error message is displayed.

(2) "Set Copy Diskeetes and Strike any key"
In response to this message, load the appropriate media to the specified units respectively and press a key other than the END and RESET keys.

(3) If the volume password is set for the source medium, the system displays the message:
"Protected by User Volume Password!"
and then the prompt:
"Input Volume Password -"
Enter the volume password in response to this prompt. If it is entered incorrectly, the system displays the following message and then returns to the message shown in Item (2):
"Unmatched Volume Password!"

(4) When the correct volume password is entered, the system starts data transfer and displays the numbers of cylinders which has been read. When reading of all cylinders is completed, the system displays the message 'Completed' and returns to the message shown in Item (2). To exit from the VCOPY command, press the END key.

[Examples]

1) VCOPY FA0: FA1:

```
Set Copy Diskettes and Strike any key
   Cilinder 74 Completed
```

This transfers the entire contents of a standard 8 inch floppy disk from Unit 0 to Unit 1. (The same format can be used for the Mark-41.

For FA:, cylinder numbers are displayed up to 74. For MA:, they are displayed up to 76.

2) VCOPY FA0: WA0:

```
ILLEGAL CLI COMMAND
```

As shown in this example, the VCOPY command cannot be used for data transfer between different types of disks.

[Error messages]

Besides system error messages, the following error messages are output during execution of the VCOPY command:

a) Illegal Diskette!
This is displayed when the source or destination medium is not an M24 medium.

b) Read Protected!
This is displayed when the source medium is read-protected (for example, the PIPS program cannot be read).

c) Protected by System Volume Password!
This is displayed when the system volume password is set for the source medium. This password cannot be cancelled.

d) Write Protected!

This is displayed when the destination medium is write-protected. If an error occurs on the destination medium, for example, if either of the error messages shown in Items a) or d), the system displays the prompt 'Copy O.K.? (Y or N). If data is to be copied, the user should press the Y key in response to this prompt. If not, the user should press the N key.

7.29 XFER

The XFER command is used to transfer data from one file (source file) to another (destination file). This command is valid also for M200 series Mark V and Mark 41 disk files. The user can change the type of a file by specifying the local switch, C or L, for the destination filename, and can copy files as they are by specifying the global switch X. Usually a file is transfered with data such as its attribute and creation date, but such data is not transferred when data is transferred to or from an I/O file.

Data cannot be transferred from any read-protected file or to any write-protected file.

[Formats]

(1) XFER/SW [S-DIR]/S [D-DIR]/D S-FIL1 D-FIL1/SW
.... S-FILn D-FILn/SW

(2) XFER/X/SW [S-DIR]/S [D-DIR]/D S-FIL1 S-FIL2
.... S-FILn

S-DIR --- Local current directory in which the source file to be transferred is registered

D-DIR --- Local current directory in which the destination file is to be registered

S-FIL --- Source file (Use of wild cards is allowed.)

D-FIL --- Destination file

Note: The difference between the above two formats depends on whether the global switch X is specified or not.

[Switches]

1) Global switches

C --- Allows the user to specify whether each source file is to be transferred or not. If the T key is pressed when a set of a source filename and a destination filename is displayed, the source file is transferred. If a key other than the T key, it is not. (Confirmation transfer)

D --- Specifies that the date of the day when the source file was transferred is to be set for the destination file. (Date of the day)
If this switch is not specified, the source file creation date this switch is not specified, the source file creation date is set for the destination file.

F --- Specifies that the source file is to be transferred even if the specified destination file has already been registered, that is, the destination file is overwritten by the source file. (Forced transfer) This processing is done on a destination file with the P attribute, but if the file has both the attributes A and P or if a password is set for the file, the destination file is not deleted.

X --- Specifies that the source file is to be copied as it is. That is, its filename, attribute, creation date, file type, etc., are registered as they are in the destination directory. (Extended transfer)

2) Local switches

C --- Specifies that the destination file is to be stored as a contiguous file. (Contiguous file)

L --- Specifies that the destination file is to be stored as a linked, that is, sequential file. (Linked file)

S --- Specifies the local current directory in which the following source filename is registered. (Source)

D --- Specifies the local current directory in which the following destination filename is registered.

[Filename]

An M200 series file is specified in the following format:

FXn:filename

- Unit number (0 to 3)
- Disk drive unit name: Mark V or 41 floppy disk (double side and double density)

The user can use a wild card when specifying source files stored on an M243 EX or M200 series disk, but it cannot be used when specifying destination files. If no destination filename is specified, the source filename is assumed.

[Display format]

"FA0:[USER1.DIR]FILE1--->WB0:[USER2.DIR]FILE2 Transferring"

When transfer of a file starts, the system displays a message in the format shown in the above example and when it is completed 'Transferred' replaces 'Transferring'.

When transferring a file from or to an M200 series disk, no directory name is displayed.

If the global switch C is specified, the system waits for a user input first without displaying 'Transferring'. If the T key is pressed at this point, a message is displayed in the format shown in the above example. If a key other than the T key is pressed, the system displays 'Skip'.

[Examples]

(1) XFER ABC.SR [USER]DEF.SR/C

This converts the file ABC.SR registered in the current default directory into a contiguous file and transfers it to the file DEF.SR registered in the directory USER within the same device. If a local current directory name is omitted as shown in this example, the current default directory is assumed to be the local current directory. For example, if the current default directory is FA0:[SYS] and the local current directory name is not specified, the file ABC.SR registered in the directory SYS is transferred to the file DEF.SR registered in the directory USER.

(2) XFER/F PROGRAM.SAV FA1: DATA FA1:

In this example also, no local current directory is specified. In the first specification, the destination file name is omitted, so the file PROGRAM.SAV is transferred to the current default directory on the device FA1: under the same file name PROGRAM.SAV. This processing is done regardless of whether the FA1: disk contains the same file PROGRAM.SAV or not, because the global switch F is specified. The same applies for the next specification of the file DATA.

If the current default directory on the source disk indicates FA1: as the current default device, an error message is displayed because the same device and the same filename are specified.

(3) XFER FA0:[USER]*.SR FA1:[ABC]

This example uses a wild card. Files registered in the directory FA0:[USER] and covered by the wild card .SR are transferred to the directory FA1:[ABC].

When a wild card is used, filenames covered by it cannot be changed, so no destination filename must be specified.

(4) XFER KB POUT

In this example, an I/O filename is specified as an output filename. Data entered from the keyboard is output to the file POUT. As an example, if the file POUT is linked to the printer MSLP-400, and the D key following the ESC key is pressed, the printer is set to the draft mode.

Any printer operates according to output codes like the above and what output codes are used depends on the printer. For details on output codes, the user should refer to the appropriate printer manual. The user can end data entry from the keyboard by pressing the CTRL and Z keys at the same time.

(5) XFER [ABC]/S FA1:[DEF]/D FILA FILD FILB FILE

In this example, two local current directories are specified. If an item is omitted in a local current directory, the required data on the item is assumed from the current default directory. In this example,

therefore, if the current default directory specifies FA0: as the current default device, the file FILA registered in the directory FA0:[ABC] is transferred to the file FILD registered in the directory FA1:[DEF] and the file FILB registered in the directory FA0:[ABC] is transferred to the file FILE registered in the FA0:[DEF].

(6) XFER [USER]/D ABC.SR DEF.SR/C

Like the example (5) above, this example specifies a local current directory. Since a local current directory for the source file is omitted, the current default directory is assumed. This XFER command performs exactly the same processing as the command written in Item (1).

(7) XFER/X/F FA1:/D PROGRAM.SAV DATA

This XFER command copies two files as they are and performs exactly the same processing as the command written in Item (2). If a file is to be transferred without any change of the filename, file type, attribute, etc., the global switch /X should be used as shown in this example. The user does not need to specify destination filenames.

(8) XFER/X FA0:[USER]/S FA1:[ABC]/D *.SR

This XFER command performs exactly the same processing as the command written in Item (3).

(9) XFER [MASTER]/S A* FA0:[A] B* WB1:[B] X* FX1:

This distributes files registered in a directory to a few directories. Among files registered in the current default directory MASTER, those covered by

the wild card A* is transferred to the directory FA0:[A.DIR], those covered by B* are transferred to WB1:[B.DIR], and those covered by X* are transferred to the FX1: local current directory (M200 series Mark-V Unit 1).

(10) XFER/X [MASTER]/D MA1:[A]A* WB1:[B]B* FX1:X*

On the contrary to the XFER command written in Item (9), this command collects files from a few directories into one directory. Files covered by these wild cards are returned to the current default directory MASTER.

(11) XFER/X [ABC]/S [DEF]/D FILA FILB [CHI]/D FILC FILD

This example shows that a local current directory is replaced with another in one XFER command. The local current directory for source files are valid through the command, but the local current directory for destination files are changed from [DEF] to [GHI]. Therefore, the files FILA and FILB registered in the directory ABC are transferred to the directory DEF and the files FILC and FILD also registered in the directory ABC are transferred to the directory GHI.

Notes:

- If an XFER command is entered under the following conditions, the destination file is overwritten by the source file:
  - The global switch /F is not specified in the command, and

- The command specifies that a source file is to be converted to a contiguous file and then transferred to the destination file, and
- A contiguous file with the same destination filename has already been registered in the destination directory.

- If no contiguous file with the same destination filename is registered, the source file is transferred under the destination filename. And if a file with the same destination filename is a sequential file, an error message is displayed.
- If a password is set for a file or directory to be accessed, the password must be specified.
- If a wild card is used for specifying source files, and a system file (attribute P) or directory file (attribute I) is covered by the wild card, the file is skipped and its filename is not displayed.

  If there is no file covered by the wild card if there is no file which can be transferred even if it is covered, an error message is displayed.

CHAPTER 8 PROCESSING PROGRAMS

Processing programs supported by MDOS/XMS are largely divided into the following four groups:

- System programs
- Utility programs
- Application programs
- User programs

The first three groups of programs are available from SORD and provided upon receipt of an order.

8.1 System Programs

MDOS/XMS supports the following system programs:

- BASIC (BASIC compiler)
- BASIC-II (BASIC for commercial use)
- SBASIC (BASIC for scientific calculation)
- RASM (relocatable assembler)
- EDIT (editor)
- DBG (debugger)
- RLDR (relocatable loader)
- LFE (library file editor)
- FORTRAN
- COBOL
- PIPS (SORD original simple programming language)
- SGL (SORD graphic language)

The BASIC compiler is provided as the standard system program of the M243EX mark series computer on a floppy disk.

8.1.1 BASIC

The BASIC language provided for the M243EX mark series computer was developed by SORD and can be executed on any SORD computer model. In recent years, BASIC has widely been used as a microcomputer-oriented language for beginners and for commercial use. SORD, taking the initiative in this trend, has developed its original BASIC language from early on and designed it to operate most efficiently on any SORD computer.

The SORD BASIC language is supported by a number of subroutines and the user can add these subroutine functions to it as required.

These subroutines are:

- Subroutine for analog data input and output
- Subroutine for digital data input and output
- Subroutine for the HC-IB I/O interface
- Subroutines for various communications

The BASIC-II program is a upgraded version of the BASIC program, and it allows a wider range of data entry formats to be used and adds many more functions. Basically, BASIC-II is suitable for commercial use. SBASIC is designed to be able to calculate a many number of digits of data more efficiently and suitable for scientific calculation.

8.1.2 RASM

The RASM program is the relocatable assembler for the M243EX mark series computer and assembles source programs written in the assembler language.

[Format]

RASM input filename 1 --- input file name n output file name file name/B output file name/L

Input file name: The input file must be registered in a directory, be a source program written according to the JIS code system, and be terminated by the assembler pseudo-instruction .END or .EOT, depending on whether the file is the last in the entry. If two or more input files are specified in one entry, files other than the last must be terminated by .EOT and the last file must be terminated by .END. Files for which the local switch B, E, or L is not set are specified as input files.

Output file name: An output file for which the local switch B is set is specified as the relocatable binary file, an output file for which the local switch L is set is specified as the listing file, and an output file for which the local switch E is set is specified as the error listing file. Different filenames must be specified for these output files.

Global switch

E --- Error inhibit
This switch inhibits output of error messages during program assembly. But error messages on the listing file, and the messages L, P, B, C, E, and S are output regardless of this switch.

Local switches

B --- Relocatable binary file
The object module is output to the file.
The default extension is .RB.

L --- Listing file
If an I/O file is specified as the listing file, data is output to the file after tabulation simulation.
The default extension is .LS.

E --- Error listing file
Erroneous statements are output to the file.
If an I/O file is specified as the error listing file, data is output after tabulation simulation.
The default extension is .ELS.

N --- No listing
If this switch is specified for an input file, the contents of the input file are not output to the listing file.

H --- Decimal location
If this switch is specified for an input file, a location address is assigned to each statement of the input file with a decimal number and its contents including the location addresses are output to the listing file.

Number --- I/O mode

If an I/O file is specified as an output file, this specifies an I/O mode for the I/O file.

If an incorrect filename is detected during execution of the RASM program, an error message is displayed and the system returns to the CLI mode.

For details on RASM, see the following manuals:

- M200 Mark Series RASM
- M243EX Mark Series MDOS/XMS System Call Handbook

8.1.3 RLDR

The RLDR program is the relocatable loader for the M243EX mark series computer and loads relocatable binary files created by the relocatable assembler RASM on to memory so as to create the save file.

User programs are allowed to use memory from Address 1600 to FFFF (hexadecimal), but since RLDR itself must be loaded to memory, the entire memory space cannot be allocated to user programs only.

[Formats]

- RLDR save file name/S load address/L start address/P input 1 --- input n output file name/A/E/M/N/number <RETURN>
- RLDR save file name/S input [overlay file name overlay input ---] <RETURN>

Save file name/S:

The switch S specifies the file as the save file.

Save file name/S/C:

If only the switch /S is specified, the save file is created as a sequential file and if both the switches /S and /C are specified, the save file is created as a contiguous file. The default file extension is .SAV.

Load address/L:

The switch /L specifies the address from which user programs are to be loaded on to memory and the address must be specified in hexadecimal. If no load address is specified, user programs are loaded immediately after the last input file on memory. If no input file is loaded on memory, they are loaded from Address 2000 (hexadecimal).

Start address/P:

The switch P specifies the address from which the RLDR program is to be loaded and the address must be specified in hexadecimal.

Input programs 1 to n:

Specify the names of one or more relocatable binary files created by the relocatable assembler and relocatable library files created by the library file editor. The default file extension is .RB. If there is no filename which has the extension .RB, a file with the specified filename is loaded. If a relocatable library file is specified as an input file, only programs indicated by external and entry names registered in it are loaded.

Input file name/X or /C:

If a relocatable library file is specified as an input file, all programs registered in the library file are loaded. If the switch X is specified, all the programs are included in a save file to be created. But if the switch C is specified, all the programs are only loaded and are not included in a save file.

Output file name/A/E/M/N:

These switches are used to specify the format in which a loading map is to be output:

/A --- External names are listed in alphabetic order.
/E --- Undefined external names are listed.
/M --- The highest load address is output.
/N --- External names are listed in the order of addresses.

Notes:

- If a undefined switch is specified, it is ignored.
- If an inconsistent switch is specified, a command line error message is displayed.

For details on RLDR, see the M200 Mark Series RASM Manual.

8.1.4 DBG

The DBG program is the debuffer for the M243EX mark series computer and is used to debug a user program, using free areas within the user program. A program is divided into blocks, and it is executed and debugged in units of a block, referencing the intermediate result (the contents of registers and memory). The corrected program can be registered in directory.

[Formats]

DBG

DBG parameter character string

The following table summarizes DBG command functions and entry formats:

Table DBG command functions and entry formats (1/5)

Function	Format	Description
Memory reference and change of its contents	ADR /	Opens memory on the specified absolute address
	n ADR /	Opens memory according to the specified offset address
	Rr /	Opens memory according to the specified pair address r =H or D or B
	Rr±n /	H' or D' or B'
	i / i ±n /	Opens memory according to the index register

Table DBG command functions and entry formats (2/5)

Function	Format	Description
Memory reference and change of its contents	. ±n /	Opens memory from the current address minus n to the current address plus n.
	<RETURN>	Opens memory on the address following the current address.
	^	Opens memory on the address followed by the current address.
	/	Opens memory on the current address (L) and the next address (H).
	<RETURN>	Closes the memory register.
	$ADR	Loads the contents of the current addresses (L) and the next address (H) on to memory.
Loading and deleting Bit Patterns (BPs)	. $B	Loads a BP on to the current address (indicated by the memory register).
	ADR $B	Loads a BP to the specified address.
	$B	Displays loaded BPS.
	n $D	Deletes the BP n (n: 0 to 7).
	$D	Deletes all BPs.
Reference of internal registers and change of their contents	$A	Displays the contents of all internal registers.
	r $A	Displays and changes the contents of the register r.
	$X	Displays and changes the contents of the register X.
	$Y	Displays and changes the contents of the register Y.
	$S	Displays and changes the contents of the stack pointer.
	$C	Displays and changes the contents of the program counter.

Table DBG command functions and entry formats (3/5)

Function	Format	Description
Reference of special registers and change of its contents	$L	Displays and changes the contents of the start address (indicated by the location register).
	$I	Displays the contents of the interrupt register (an internal register).
	$T	Displays and changes the contents of the trace register. T=0: The contents of any register are not displayed when a BP is read. T=1: The contents of all registers are displayed when the last BP is read. T='0A: The contents of all registers are printed at the printer each time a BP is read. T=n: The contents of all registers are displayed on the screen each time a BP is read.
	$N	Displays and changes the contents of the number register. N=0: Data is displayed in hexadecimal. N='0A: Data is displayed in ASCII codes. N-n: Data is displayed in decimal.
	$0	Displays the contents of all offset registers.
	n $0	Displays and changes the contents of the specified offset register.

Table DBG command functions and entry formats (4/5)

Function	Format	Description
Search	s :xx	Searches for the address which contains data xx from the last opened address and opens memory on the address.
Program start and restart	$R	Executes the program from the address indicated by the location register.
	ADR $R	Executes the program from the specified address.
	$P	Executes the program again from the address on which the current BP exists.
	n $P	Executes $P n times.
Memory dump	ADR1,ADR2T	Dumps the contents of memory from ADR1 to ADR2 to the screen.
	ADR1,ADR2L	Dumps the contents of memory from ADR1 to ADR2 to the printer.
	ADR1:nT	Dumps n bytes of memory contents from ADR1 to the screen.
	ADR1:nL	Dumps n bytes of memory contents from ADR1 to the printer.
Operation	Expression=	Displays the result of the expression. (Two bytes operation of additon and subtraction)
Memory padding	ADR1,ADR2,xxP	Loads data xx on to memory from ADR1 to ADR2.
	ADR1:n,xxP	Loads n bytes of data xx on to memory from ADR1.

Table DBG command functions and entry formats (5/5)

Function	Format	Description
I/O	RP port No.	Reads data from the specified port.
	WP port No,data	Writes data into the specified port.
Program loading functions	M $filename	Loads the specified save file from the disk.
	V $filename/SW	Saves the program on the disk under the specified filename.
	V $filename/SW Adr1,Adr2,Adr3	Adr1: Load start address (Leading load address) Adr2: Load end address Adr3: Program start address No switch: Creates the save file as a sequential file. Switch C: Creates the save file as a contiguous file. Note: If a save file with the same filename already exists on the disk and the file is a contiguous file, the file is overwritten.

For details on DBG, see the M200 Mark Series RASM Manual.

8.1.5 LFE

The LFE program is the library file editor for the M243EX mark series computer and is used to edit a relocatable binary file created by RASM and a relocatable library file (a collection of relocatable binary files).

[Formats]

```
LFE old library name new library name/O
LFE listing file name/L/number
```

- Global switch --- None
- Local switches

  L ------ Listing file
  O ------ Output library
  None --- Master library

Table LFE commands

Command	Function
L filename	Lists the contents of the specified file.
No switch	Lists all titles, entry names, external names, and comments.
/A	Lists all items.
/E	Lists all titles and entry names.
/T	Lists all titles.
/X	Lists all titles and external names.
I filename	Inserts data.
C program name	Copies programs.
No switch	Copies all programs up to the specified one.
/B	Copies all programs registered before the specified one.
S program name	Skips programs.
No switch	Skips all programs up to the specified one.
/B	Skips all programs registered before the specified one.
R program name	Replaces the specified program.
D program name	Deletes the specified program.
/comment	Inserts a comment.
E	Terminates the LFE program.

Note: A program name is a name specified by TITLE.

For details on LFE, see the M200 Mark Series RASM Manual.

8.1.6 EDIT

The EDIT program is the text editor for the M243EX mark series computer and is used to create source programs such as assembler, FORTRAN, and COBOL programs, and to create command string files.

[Formats]

EDIT

EDIT input file name output file name

- Global switch --- None
- Local switch

  T --- If an I/O file is specified as an output file, data is output after tabulation simulation.

EDIT commands are divided into the following six groups:

- EDIT commands for floppy disks
- EDIT commands for I/O operations
- EDIT commands for editing
- EDIT commands for the CP
- EDIT commands for deletion of character strings and lines
- Macro commands

Table EDIT commands for floppy disks (1/2)

Command	Function
CR (filename) $$	Specifies the file to be edited. Data is read from the specified file to memory, edited, and stored again on a disk under a different filename. To edit more than one file, the user must enter this command for each file. When a GR command is entered, the previous file is closed and a new file specified by the command is opened.
CR (filename)/T$$	The switch T specifies that all FF (16) codes in the specified file are to be ignored.
GW (filename) $$	• Stores data entered from the keyboard in the specified file on the disk. • Stores an edited file in the specified disk file. The edited file was read to memory by a GR command. • Data is read from a file to memory, edited, and stored in the disk file specified by the GW command. To create more than one file, the user must enter this command for each output file. That is, to enter a GW command, the previous GW command must be terminated by the GC command.
GW (filename)/T$$	• The switch T specifies that tabulation simulation is to be made on the specified file. This switch is valid only when an I/O file is specified as an output file.
GC$$	Closes a file used by the GR or GW command. This command must be executed when use of a file specified by the GR or GW command is completed. When a GC command is executed, a file specified by the GW or GR command is closed.

Table EDIT commands for floppy disks (2/2)

Command	Function
GL$$	Lists all filenames registered on the floppy disk.
GL/1/A$$	These switches have the same function as those for the LIST command except that the sort function cannot be used.
GD (filename) $$	Deletes the specified file.
GN (filename 1)	Changes the name of a file from 1 to 2.
$ (filename 2) $$	The GN command has the same function as the RENAME command.
EDIT <RETURN>	Loads the EDIT program into the main memory. Ready EDIT <RETURN>
EDIT (input file name) { (output file name) } <RETURN>	* ← This is the editor prompt.
H$$	Transfers control from the EDIT program to MDOS/XMS.

Note: S represents that the END key is to be pressed.

Table EDIT commands for I/O operations (1/2)

Command	Function
I (character strings) $$	Writes data into the edit buffer, that is, all character strings entered until $$ is entered to terminate the command are written into the edit buffer. A RETURN key code is also entered as a character. Data is stored in a file in 8 bit JIS codes, so English letters and katakana characters can also be stored. Data is written into the edit buffer from the CP position.
A$$	Reads data from a disk file specified by the GR command into the edit buffer continuously without clearing the buffer each time one page of data is read. Usually, when one page of data is read, data input stops.
P$$	Writes data from the edit buffer into the file. Data is written into the file in units of a page by inserting a separator between them.
nP$$	Writes n lines of data from the edit buffer, starting at the CP position, into the file.
Y$$	Clears the edit buffer and reads one page of data from the disk file.
R$$	Executes a P$$ command followed by a Y$$ command. That is, this command has both the functions of the P$$ and Y$$ commands.
nR$$	Executes the R$$ command n times.
T$$	Displays the entire contents of the edit buffer on the screen regardless of the CP position.
nT$$	Reads n lines of data from the edit buffer, starting at the CP position, and displays the data on the screen.

Table EDIT commands for I/O operations (2/2)

Command	Function
E$$	Writes part of the file specified by the GR command from the edit buffer into the file specified by the GW command and then transfers the remaining file contents to the latter file.

Notes:

- n must be a decimal number.
- Character strings are part of data being input or edited.
- The CP is short for Character Pointer.

Table EDIT commands for editing (1/2)

Command	Function
C (character string 1) $ (character string 2) $$	Searches for the character string 1 from the current CP position and when it is detected, replaces it with the character string 2. (This is done only inside the edit buffer.)
nC (character string 1) $ (character string 2) $$	Executes the C command n times.
S (character string) $$	Searches for the specified character string in the edit buffer, starting at the current CP position. The CP moves to the character string position. (This is done only inside the edit buffer.)
nS (character string) $$	Executes the S command n times.
Q (character string) $$	Searches for the specified character string in the edit buffer, starting at the current CP position. • When it is detected, the CP moves to the character string position. • When it is not detected, the edit buffer is cleared, the next one page of data is read from the disk file, and the specified character string is searched for again. This processing is repeated until the character string is detected.
nQ (character string) $$	Executes the Q command n times.

Table EDIT commands for editing (2/2)

Command	Function
N (character string) $$	Searches for the specified character string in the edit buffer, starting at the current CP position. If it is detected inside the edit buffer, the CP is moved at the end of the character string. If it is not detected inside the edit buffer, the current contents of the edit buffer are written into the file specified by the GW command, the next one page of data is read from the file specified by the GR command, and the specified character string is searched for again. This processing is repeated until the character string is detected.
nN (character string) $$	Executes the N command n times.

Table EDIT commands for control of the CP

Command	Function
B$$	Positions the CP at the head of the edit buffer.
Z$$	Positions the CP at the end of the edit buffer.
nJ$$	Positions the CP at the head of the nth line counted from the first line of the edit buffer. (This is done regardless of the current CP position.)
nL$$	Moves the CP by n lines from the current CP position. (A negative number can be specified.)
nM$$	Moves the CP by n characters from the current CP position. A RETURN key code is counted as one character. (A negative number can be specified.)
.$$	Displays the line number on which the CP is positioned in decimal, counting from the first line of the edit buffer.

Table EDIT commands for deletion of character strings and lines

Command	Function
nD$$	Deletes n characters from the current CP position.
nK$$	Deletes n lines of data from the current CP position.
:$$	Displays the number of lines of which data exists in the edit buffer in decimal.

Table Macro commands

Command	Function
XM (command string) $$	Defines a macro which can be specified with up to 256 characters (including the END key code as one character) and deletes the previously defined macro.
XT$$	Displays the contents of the current macro.
XD$$	Deletes the current macro.
X$$	Executes the currently defined macro command string.
nX$$	Executes the X$$ command n times.

For details on EDIT, see the M200 Mark Series Manual.

8.1.7 FORTRAN

The FORTRAN language is suitable for calculating expressions and handling numerical data. FORTRAN is an abbreviation for the FORmula TRANslator and the Japan Industry Standards (JIS) establishes the standard FORTRAN language system. It is divided into three levels: Level 3000, 5000, and 7000. FORTRAN provided by SORD has functions equivalent to Level 7000 except that it does not handle complex numbers.

When executing FORTRAN on the M243EX mark series computer, note the following points:

- If no printer name (Device No. 2) is specified, the serial printer (SOUTA file) is assumed.
- To use the parallel printer (POUT file), it must be specified by the following statement and as a result, the printer (Device No. 2) is set to the parallel printer:

```
CALL OEPN(2'OUT '0)
```

[Format]

```
CALL OPEN(2'POUT '0)
```

Relocatable linker

The relocatable linker allows the user to set a directory name and a password for each file.

The following switches functioning as commands to the relocatable linker (hereafter called LINK) are added or were changed:

[Additional switches]

L: Outputs all global symbols to the printer.

C: Switches the printer to which global symbols are to be output from serial (SOUTA) to parallel (POUT).

I: Switches the printer to which global symbols are to be output from parallel (POUT) to serial (SOUTA).

The printer is set to the serial printer (SOUTA) at initialization of LINK.

[Changed switches]

R: Returns LINK to the initial state.
All programs loaded before this switch is specified are deleted, all switches specified before this switch are made invalid, and LINK is set back to the initial state.

E: Creates a save file.
When linking an object file (relocatable binary file) which has been notified of a run time library name by the compiler or assembler, the run time library is retrieved and the undefined symbol program is loaded. Then the save file is created. In this case, the run time library name (such as FORLIB or COBLIB) must be registered in the system directory SYS.DIR of the disk from which LINK was called, or of the system default disk from which MDOS/XMS was bootstrapped. Generally, a run time library name is registered in the same directory in which the linker, compiler, and assembler are also registered.

LINK automatically adds the file extension .SAV to the created save file.

G: Executes the program created by specifying the switch E.

The following table shows files required for execution of the FORTRAN program:

Table Files necessary for FORTRAN execution

Directory name	File name	Function
SYS .DIR	EDIT .SAV	Text editor
"	FORT .SAV	FORTRAN compiler
"	LINK .SAV	FORTRAN linker
"	FORLIB.LIB	FORTRAN run time library

For details on the FORTRAN compiler, see the following manual:

FORTRAN-80 (GEJ-0130)

8.1.8 COBOL

COBOL is an abbreviation for the COmmon Business Orinted Language and as it says, it is a business-oriented language. COBOL provided by SORD is named 'COBOL-74' and conforms to the American National Standard (ANSI) X.3.23 1974.

When executing COBOL on the M243EX mark series computer, note the following points:

- 256 bytes are handled as one block so as to process disk files more efficiently.
- The file sharing mode is used when opening a file in the input mode. The file monopolization mode is used when opening a file in a mode other than the input mode.

Print file

To be able to handle various printers as print files, printer names are externally defined by the sub-program for specifying a printer name, LPNAME. As a result, when a sub-program name associated with a printer is specified, the sub-program is automatically loaded and the printer is assigned as the print file.

The following five sub-programs are registered in the system directory of the disk which contains COBOL and they are associated with output channels for I/O devices as follows:

Sub-programs		Output channels for I/O devices (printer names)
PSOUTA.REL	----------	SOUTA
PSOUTB.REL	----------	SOUTB
PSOUTC.REL	----------	SOUTC
PSOUTD.REL	----------	SOUTD
PPOUT. REL	----------	POUT

Note: These sub-programs are not registered in the directory IO.DIR and output channels for I/O devices cannot be used.

To specify the above sub-programs associated with the respective output channels connected to printers, the XFER command must be used. When a sub-program name is specified in an XFER command, the contents of the sub-program is transferred to the sub-program for specifying the printer name, LPNAME.REL.

[Examples]

SOUTB can be specified in the following format:

```
XFER [LINKLP[PSOUTB.REL [SYS]LPNAME.REL
```

This XFER command transfers the contents of the sub-program PSOUTB.REL to the printer name specification sub-program LPNAME.REL and assigns the I/O output channel file as the printer file. As a result, when a relocatable binary file created by the COBOL compiler is linked by executing the program LINK.SAV, the save file is created in the SOUTB print file.

To switch the print file from SOUTB to POUT, enter an XFER command as follows:

```
XFER [LINKLP]PPOUT.REL [SYS]LPNAME.REL
```

When this command is entered, COBOL save files created by LINK.SAV are written into the print file POUT.

The LPNAME.REL sub-program is automatically loaded by the linker during LINK.SAV execution.

FLDUMP.SAV

The M243EX mark series file dump program FLDUMP is a revised version of the M203 and M223 mark series FLDUMP program and it has the new function described below.

The user can specify a variety of printers as the output file. If the printer is specified as an output file, that is, P is entered, the system displays the following message and waits for a user input:

```
PRINTER NAME:
```

In response to this message, the user must enter the name of a printer (output channel name) to which data is to be output, a slash as a separator, and a mode number for the printer. If the mode number is omitted, the mode number 3 is automatically assumed.

[Example]

```
OUTPUT FILE (C =CRT, P=PRINTER) :P
PRINTER NAME:POUT/1
```

Except this function, this FLDUMP program has exactly the same functions as the M203 and M223 mark series'.

For details on COBOL, see the M200 Mark Series COBOL-74 Manual.

8.1.9 PIPS

PIPS is an abbreviation for the Pan Information Processing System and is a simple programming language provided by SORD.

PIPS does not require program creation, and can be built into user applications at it is and operated on any SORD computer model.

PIPS has been improved regularly and many functions such as that for displaying business graphs have been added so far.

8.1.10 SGL

SGL is an abbreviation for the Sord Graphic Language and is a graphic language provided by SORD. It is the standard SORD graphic language and can operate on any SORD computer model. By using SGL, the user can establish a graphic system more easily. The standard M243EX also supports the SGL. To use SGL, the user needs to specify use of SGL by one of CLI commands, SGL, at system generation.

For details on use of the SGL command, see the following manual:

SGL (GD09J-0018-1)

8.2 Utility Programs

8.2.1 INIT

The INIT program is used to initialize the medium in the specified unit physically, that is, format it. Any medium other than SORD's cannot be used on the M243EX system. So, to use such a medium on the M243EX system, it must be formatted and its system areas such as the system directory area must be initialized beforehand. For these purposes, the INIT command is used.

Any initialized medium can be used as a data medium, but, to use such a medium for MDOS/XMS, the user needs to register operating system files and required CLI command files by using the OSCHG command.

This command is valid only for floppy disks.

[Format]

```
INIT device number:
```

[Switch]

None

[Examples]

1)
```
INIT WA:
ILLEGAL COMMAND
```

2)
```
INIT FA1:
Floppy Disk Initialize Program Rev. 00D
  SECTOR SEQUENCE: 05
  START INIT ? YES(Y) OR NO(N) OR END(E) Y 00
  START INIT ? YES(Y) OR NO(N) OR END(E) E
READY
```

3) INIT FA1:
```
Floppy Disk Initialize Program Rev. 00D
  SECTOR SEQUENCE: 08
  START INIT ? YES(Y) OR NO(N) OR END(E) Y 76
SD:0 CY:76 01  WRITE FAULT ! 75
```

In Example 1), a winchester fixed disk is specified as the device number. This command cannot initialize any winchester fixed disk, so the error message 'ILLEGAL COMMAND' is output.

The INIT command shown in Example 2) initializes the medium in the floppy disk drive unit 1. When an INIT command is entered, the INIT program announces itself as shown in the example, displays the message 'SECTOR SEQUENCE:', and waits for a sector sequence number to be entered. The sector sequence number must be from 1 to 13. For the M243EX system, usually the sector sequence number 8 is assigned to FA:, but if the disk in FA: can be accessed more efficiently by assigning a sector sequence number other than 8, the number should be assigned. That depends on the file access method of each application program. The user can specify the sector sequence number 8 only by pressing the RETURN key. When a sectory sequence number is specified, the following message is displayed:

```
START INIT ? YES(Y) OR NO(N) OR END(E)
```

In response to this message, load a medium to be initialized in the specified drive unit and type Y to start initialization.

When Y is typed, cylinder numbers of the medium following Y are displayed. These numbers start with 76 and end with 0.

When the cylinder number 0 is displayed and initialization is completed, the following message is displayed again:

```
START INIT ? YES(Y) OR NO(N) OR END(E)
```

To initialize another medium, replace the initialized medium with a new one and type Y. To finish initialization, type E. The system returns to the READY prompt. To change the sector sequence number at this point, type N. The message 'SECTOR SEQUENCE:' is displayed and the user can enter a sector sequence number.

In Example 3), an error occurs during initialization. If an error occurs during execution of an INIT command, data on the error is displayed like this:

```
SD:0 CY:76 01 WRITE FAULT! 75
```

SD:0 Indicates the side of the medium on which the error occurred.

CY:76 Indicates the number of the cylinder on which the error occurred.

01 Error status. This is a hexadecimal number.

```
BIT 7 DRIVE NOT READY
BIT 0 TIME OUT ERROR
```

WRITE FAULT! Indicates that the medium was initialized abnormally.

The system then continues initialization and displays the numbers of initialized cylinders. To stop initialization, press the RESET or HALT key. The system returns to the following message:

Notes:

(1) Any medium must be initialized carefully. If a medium was initialized in a drive unit and an attempt is made to access the medium in a drive unit other than that used in the initialization, it may fail. To avoid such an error, SORD sets a standard drive unit to be used for initialization.
The user also should not use any drive unit other than that set to be used for initialization.

(2) Do not make any attempt to access a medium being initialized. Basically, the INIT command must be executed singly.

(3) Some MDOS/XMS versions do not allow use of the INIT command. If an INIT command is executed under control of such an MDOS/XMS version, the system displays the following message and returns to the READY prompt:

```
OS VERSION ERROR
```

8.2.2 SIFT

The SIFT program is used to convert the format of a 8 inch floppy disk file.

That is, the SIFT program converts an IBM format file into the SORD format file and transfers it to the M243EX mark series system. On the contrary, it converts a SORD format file into the IBM format file and transfers it to an IBM system. The user can change the name, attribute, creation date, and effective date of any data file written in the IBM format, and displays a list of data set names, and change the code system by this SIFT program.

It also allows the user to transfer a file from a single-side and single-density floppy disk to a double-side and double-density floppy disk.

Number of recording sides	Density	Number of bytes per sector
1	Single	128
1	Single	256
1	Single	512
2	Single	128
2	Single	256
2	Single	512
2	Single	256
2	Single	512
2	Single	1024

Note: Records must not be blocked or spanned, or extends over a single voluem.

Fig. Standard IBM floppy disk recording formats which can be handled by the SIFT command

The SIFT program can be activated by typing SIFT and pressing the RETURN key in the CLI mode.

Table SIFT commands

Command name	Function
LIST	Displays a list of data set names.
IFER	Transfers the specified file from the IBM floppy disk to the M243 system after conversion.
SFER	Transfers the specified file from the SORD floppy disk to the IBM system after conversion.
ERR	Displays the contents of erroneous tracks recorded in the error map.
CHATR	Changes the attribute of the specified IBM floppy disk file.
CHANG	Changes the code system from JIS to EBCDIC or from EBCDIC to JIS.
DATE	Changes the creation and effective dates of the specified file.
NAME	Changes the specified IBM floppy disk file name.

Differences between M223 and M243 SIFT programs

The M243 mark series SIFT program differs from the M223 mark series' in the following points:

- It has no INIT command.
- It does not allow the record or block length specified in the SFER command to exceed the number of bytes per sector of the IBM floppy disk. It recognizes each floppy disk type, so the user does not need to specify it.

- It outputs any message in kana characters, which cannot be changed even by the SYSTEM command.
- It accepts M243 floppy disk file names specified in the standard MDOS/XMS file specification format shown in the following example:

  Example: WA0 : [SIFT]DATA
           FA1 : SIFT. DAT

- It is terminated if an erroneous track is detected.
- It assumes that the printer is linked to SOUTA.

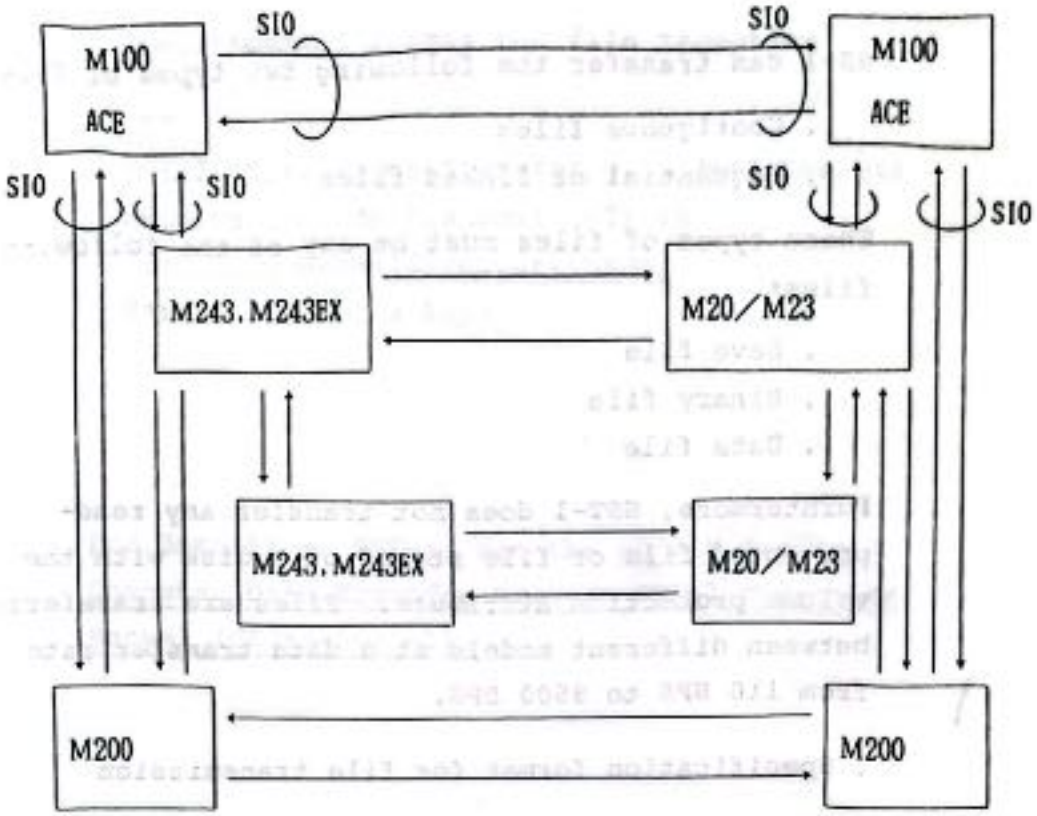
For details on SIFT, see the following manuals:

- M200 Computer Series Utility Programs No. 4
- 8 Inch Floppy Disk File Converter Program Operation Guide

8.2.3 SST-1

The SORD Super Transfer - 1 (SST-1) program is used to transfer files between SORD computer systems through the RS232C interface.
SS-1 supports the following models:

- M200ACE
- M20 and M23
- M200 (including the mark series)
- M243 and M24



Note:

M100ACE ..... SIO

M20 and M23 mark series ... RS232C B channel

M200 ..... RS232C B channel

M203 and M223 mark series.. RS232C B channel

M243 and M243EX mark series RS232C B channel

Fig. SST-1 support models

SS-1 can transfer the following two types of files:

. Contiguous files
. Sequential or linked files

These types of files must be any of the following files:

. Save file
. Binary file
. Data file

Furhtermore, SST-1 does not transfer any read-protected file or file stored on a disk with the volume protection attribute. Files are transferred between different models at a data transfer rate from 110 BPS to 9600 BPS.

. Specification format for file transmission

```
SST
SST-1 (Sord Super Transfer - I) Revision 01F
Receive (=R) or Transmit (=T) :T
File Name:filename-1 filename-2 .... filename-3
Strike the space key:
```

:
:
:
:

The user enters underlined data from the keyboard.

. Specification format for file reception

```
SST
SST-1 (Sord Super Transfer - I) Revision 01F
Receive (=R) or Transmit (=T) :R
Directory Name:[directory name]
Strike the space key:_
```

:
:
:

For details on SST-1, see the SORD M Series Computer Data Transfer Program SST-1 Operation Manual (GE09J-0015-1).

8.2.4 TAPFER

The TAPFER program is used to transfer data from the paper tape reader/puncher (HRP-250S) to a disk and vice versa through the SIO.
The TAPFER program can be activated by specifying a TAPFER command in the following format like other CLI commands:

. Data transfer from the paper tape reader to a disk
  TAPFER/SW SINB disk file name

. Data transfer from a disk to the paper tape puncher
  TAPFER/SW disk file name SOUTB

The user can specify either of the following two global switches for the TAPFER command:

Global switches

None ---- Specifies binary data and that data check is not to be made.

A ------ Specifies ASCII data and that even parity check is to be made. Data is transferred to the disk, removing parity bits.

Examples

. Transferring data from the paper tape reader to a disk

  If the power switch of the paper tape reader/puncher HRP-250S connected to the computer is turned on and it is loaded with paper tape, the paper tape is just fed and cannot be read normally. To avoid this, paper tape must be loaded according to the following procedure:

Step 1: Turn the HRP-250S POWER switch on.

Step 2: Enter a TAPFER command from the keyboard as follows:

```
READY
TAPFER SINB disk file name
```

READY is a system output message. The disk file name must be the file of a disk file into which data is to be written from the paper tape. The disk file must be new or a contiguous file created by the CREATE command. If the specified filename already exists, the following message is displayed on the screen:

```
FILE ALREADY EXIST
```

Step 3: When the TAPFER command is entered correctly, the following message is displayed:

```
SET PTE AND STRIKE ANY KEY
```

Step 4: In response to this message, load the paper tape into the HRP-250S and press any key, thus strating reading of the paper tape.

Step 5: When a timeout error occurs, that is, when no data is read for two to three seconds, it is recognized as the end of the tape. Therefore, portions which do not contain any data or contains unnecessary data should be removed from the paper tape beforehand. If the tape ends, the READY message is displayed on the screen.

Step 6: Turn the HRP-250S POWER switch off.

. Transferring data from a disk to the paper tape puncher

First of all, when transferring data to paper tape, check that the HRP-250S is loaded with paper tape enough to punch data of the file to be specified and then follow the procedure described below:

Step 1: Turn the HRP-250S POWER switch on.

Step 2: Turn the HRP-250S PUNCH switch on.

Step 3: Enter a TAPFER command from the keyboard as follows:

```
READY
TAPFER disk file name SOUTB
```

The disk file name is the name of a file from which data is to be transferred to the paper tape puncher. When the TAPFER command is entered correctly, output to the tape puncher starts. When the output is completed, the READY message is displayed on the screen.

Step 4: If there are two or more files to be transferred, the user must repeat Step 4 for each file.

Step 5: Turn the HRP-250S PUNCH switch off.

Step 6: Turn the HRP-250S POWER switch off.

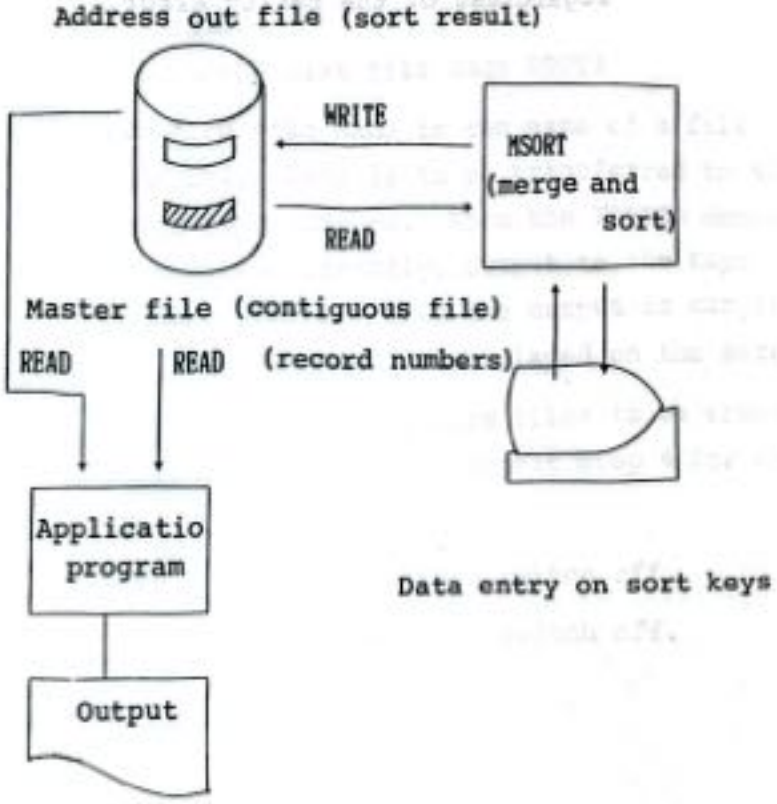
Precaution

The filename SOUTA used in the above example is valid only when the HRP-250S is connected to the SIO channel B (Standard B channel specifications). If it is connected to the channel A, the two SIO filenames, SINA and SOUTA, must be specified.
If a parity check error is detected while ASCII data is being read from paper tape, the TAPFER program displays the message 'PARITY ERROR' and converts the erroneous data from paper tape in units of 2K bytes. The message 'PARITY ERROR' indicates that one or more parity errors exist in the 2K bytes of data. TAPFER continues data transfer to theend of the paper tape regardless of the parity error.

8.2.5 MSORT

The Multikey SORT (MSORT) program is used to sort and merge data records in a file created by the BASIC program, using data (numerical and character data) as sort keys.

The MSORT program allows the user to specify up to five sort keys, using numerical data items and character strings in data records and arranges these data records in ascending or descending order. Furthermore, it outputs a sort and merge result containing key data and record numbers used in the input file to a specified file. Figure 8.5 shows how MSORT sorts data records in a contiguous file.



Concept of multikey sort

The MSORT program has the following specifications:

. Maximum number of keys: 5

. Key data types:

Single-precision numerical data, double-precision numerical data, and character data handled by BASIC and CBASIC. These three types of data can be mixed in a record to be sorted.

. Key data size:

Maximum 170 bytes. This is the total number of bytes of key data.

. Record length:

Maximum 10K (10,240) bytes

. File size:

Maximum 8M bytes

. File type:

Contiguous type only

. Deletion function:

If specific data (hexadecimal 'FF) exists at the specified position in a data record to be sorted, the record is not sorted.

. Sort format specification:

Input file, output file, and work file names, and keys can be specified from the keyboard. These parameters can be registered on a medium so as to be able to do regular jobs more easily.

. I/O formats for disks:

Input: Data is input in units of a record.

Output: When sorting ends, key data and record numbers used in the file before the sorting are output to the specified file. MSORT transfers data to I/O devices in units of a block to make it at a high speed.

The MSORT program can be activated by entering any one of the following MSORT commands in the CLI mode like other CLI commands:

MSORT

When this MSORT command is entered, the program announces itself on the screen. It allows the user to specify all items on sort keys from the keyboard interactively.

MSORT filename

When this command is entered, the specified parameter save file is called and data on sort keys is automatically loaded.

MSORT/G filename

When the switch G is specified as shown in this example, the specified parameter save file is read, data on sort keys are loaded, and sorting is started automatically.

JOB MSORT/G filename

When this MSORT command is entered, the same processing as the above MSORT/G command is performed. But, since a JOB command is specified, sorting is done as a background job.

Table 8.23 shows sort and merge control information to be specified interactively when executing an MSORT command.

Table Sort and merge control information

Sort and merge control information	Input format (example)	Constraing
1. Source file: (The name of file to be sorted)	1.:DATASR	Disk file only
2. Out file: (The name of a file to which the sort result is to be output)	1 :DATADT	Disk file only
3. Work file: (The name of work file to be used for sorting)	DATAWK	Disk file only. The source or output file specified above cannot be specified.
4. Record total: (number of records to be sorted)	500	The number of records must be from 1 to 65,535.
5. Record format:		
Record length: (The length of one record)	35	The record length must be in the range from 1 to 10,240 bytes.
Delete position: (A number or the RETURN key only if there is no specific data)	2 or the RETURN key only	The number must be in the range from 1 to 10,240.
Key 1,2,...., 5	Enter key data in the format a,b, and c.	
6. Sort format save file: (The name of a file in which these parameters to be saved or the RETURN key only if they are not to be saved)	TABLE	Disk file only. If the specified filename already exists, an error message is displayed.

For details on the MSORT program, see the M243 Computer Series Utility Program Multikey Sort Program Operation Manual.

8.2.6 SYSGEN

The M243EX mark series MDOS/XMS operating system allows the system to be expanded by increasing the number of disk drive units such as FA, WA, WB, and WC, and the number of terminals, and by extending the main memory. The user can specify these devices and other functions to be added by using the SYSGEN program according to the user needs.

SYSGEN allows the user to:

- Specify the CCU-10.
- Specify the number of jobs.
- Specify disk drive unit types to be used and the number of disk drive units.
- Specify the number of system channels.
- Specify the number of memory files.
- Specify a user bootstrap message.
- Specify the following for the serial ports A, B, C, and D respectively:
  - Whether the serial port is to be used as an I/O port (SIN/SOUT) or to be connected to a terminal, or that it is not to be used
  - an initial baud rate
- Create a function table for each console and terminal.

SYSGEN operating environment

MDOS/XMS can be largely divided into the program portion and buffer table portion. There are various buffer tables and they differ in size and contents. These buffer tables required by the SYSGEN program are stored on one memory area, which is called the work area.

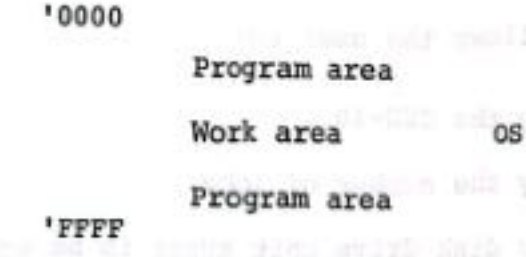


Fig. M24 MDOS/XMS operating system area format

Buffer tables prepared according to user specification are loaded into the work area at system generation. The size of the work area is fixed, so it may become insufficient, depending on the user specification. If it occurs, the message 'WORK AREA FULL' is displayed and the user needs to specify the above parameters again.
In the MDOS/XMS version 31A, about 6.8K bytes of the work area is provided. For details on the number of bytes to be reserved for buffer tables, see the section entitled 'Input request messages' written later.
Note that the size of the work area varies with the operating system type and version.

To make system generation, follow the procedure described below:

1. Bootstrap MDOS/XMS from a medium which contains SYSGEN.SAVE.
2. Log into the system to enter the READY state.
3. Check that any other job is not executed and load the SYSGEN program.

   ```
   READY
   SYSGEN [filename[/SW]]
   ```

   If a filename is specified, parameters specified by the user written into the file after system generation is completed. If an I/O file name is specified as the filename, the switch (mode number) must be specified for the device.
4. SYSGEN searches for a free memory bank and if it is detected, SYSGEN reads MDOS/XMS from the medium and write it into the memory bank.
5. Generations items and their default values are displayed on the screen.
   These default values are not those defined in MDOS/XMS bootstrapped from the medium and were just set when the system was shipped from a SORD factory. The same default values are always displayed for the same SYSGEN version.

```
*** M24 MDOS/XMS SYSTEM GENERATION *** (Rev. 21A)

CCU ------------- N        memory boards ------ 0
jobs ------------ 2        disk units --------- FA2 WA0
                                                 WB0 WC0
system channels -- 16       memory files ------- 10
user boot message-
code conversion -- SORD COD

                                                    Function
          Baud   Mode    Parity  Stop-bit  Data (T/R)  I/O    Conv.  BASIC.FNC
console-
SIO A -- 9600    wait      non      2        8/8      SIN/
                                                     SOUT
SIO B -- 9600    ignore    non      2        8/8      SIN/
                                                     SOUT
SIO C -- 9600    wait      non      2        8/8      RT-20
SIO D --                                             not-
POUT ---                                             used

-------------------------------------------------------------------------
          RETURN -------default value                    LEFT:10003
          END (CTRL/W)--no function                      USED:     0
-------------------------------------------------------------------------

     CCU YES OR NO?
```

Fig. Data displayed at activation of MDOS/XMS SYSGEN

6. Following the above data, the system displays input request messages one after another on the screen. In response to these messages, the user needs to enter data from the keyboard. EAch input request message line must be terminated by pressing the RETURN key. Entered data is underlined and replaces the default value at the same position.
   If only the RETURN key is pressed, the default value is assumed. Or if only the END key is pressed or if the CTRL and W keys are pressed at the same time, the parameter item is not supported by MDOS/XMS. The message 'LEFT: XXXXX USED: XXXXX' is also displayed at the lower right-hand side of the same screen and it indicates the status of the work area. That is, LEFT indicates the number of bytes which are not used in the work area and USED indicates the number of bytes used.

7. When all data is entered in response to these input request messages, the newly generated MDOS/XMS is written into the operating system file of themedium.
   Never reset system hardware or turn the system power off at this point.

8. If a SYSGEN file name is specified, data on the system generation is recorded in the file. And if it is a disk file, the user can display the contents of the file by using the editor. To display its contents, enter an editor Y$$ command followed by a T$$ command. When the Y$$ command is executed, a question mark (?) is displayed, but do not worry. This is because no page end code ('0C') was not written into the file. Data is being read normally.

9. When writing of the new MDOS/XMS is completed, the system returns to the READY prompt. The user can use the new MDOS/XMS by bootstrapping it from the medium and overwriting the old MDOS/XMS on Bank 0.

Input request messages

Input request messages displayed by SYSGEN and their meanings are as follows:

1. "CCU YES OR NO?"

   Specify whether the SORD communication controller CCU-10 is to be used. If it is to be used, type Y and if it is not to be used, type N.
   When Y is typed, MDOS/XMS is set to support the CCU-10 and prepares two I/O files, CCU and CCUC.

   Default value: N
   Monopolized work area size: 1,143 bytes

2. "How many optional memory boards?"

Specify the number of memory boards to be added.

Default value: 0
Monopolized work area size: 332N plus 334 bytes
(N: Number of memory boards to be added)

3. "How many jobs?"

Specify the number of jobs allowed to be executed concurrently with a number from 1 to 4. Alog-in operation from a console or terminal is counted as one job.
For example, if a user logs in from a console, two background jobs are executed, and another user logs in from a terminal at the same time, the number of jobs is counted as four.

Log-in operation from a console	1
Background jobs	2
Log-in operation from a terminal	1

The number indicates the maximum number of jobs which can be executed in parallel at the same time. If two or more jobs are executed serially, the number of jobs is one.

Each job requires one memory bank, so if there is not an enough number of memory banks to execute jobs of the number specified by this parameter concurrently, an error message is displayed. The number of memory banks is determined by the number of additional memory boards specified in Item 2 above. That is, the number of memory banks is 4N plus two. N indicates the number of additional memory boards.

Default value: 2

Monopolized work area size:
687N plus 809 bytes (N: Number of jobs)

4. "How many FA units?"

Specify the number of standard 8 inch floppy disk drive units FA to be connected with a number from 0 to 4.

Default value: 2

Monopolized work area size:
37N plus 1,153 bytes (when the number of units, N, is 1, 2, 3, or 4) or 0 (when the number of units is zero)

"How many WA units?"

Default value: 0

Monopolized work area size:
37N plus 1,153 bytes (when the number of units, N, is 1, 2, 3, or 4) or 0 (when N is zero)

"How many WB units?"

Specify the number of disk drive units WB to be connected with a number from 0 to 4.

Default value: 0
Monopolized work area size:
37N plus 1,153 bytes (when the number of units, N, is 1, 2, 3, or 4) or 0 (when N is zero)

"How many WC units?"

Specify the number of disk drive units WC to be connected with a number from 0 to 4.

Default value: 0
Monopolized work area size:
37N plus 1,153 bytes (when the number of units, N, is 1, 2, 3, or 4) or 0 (when N is zero)

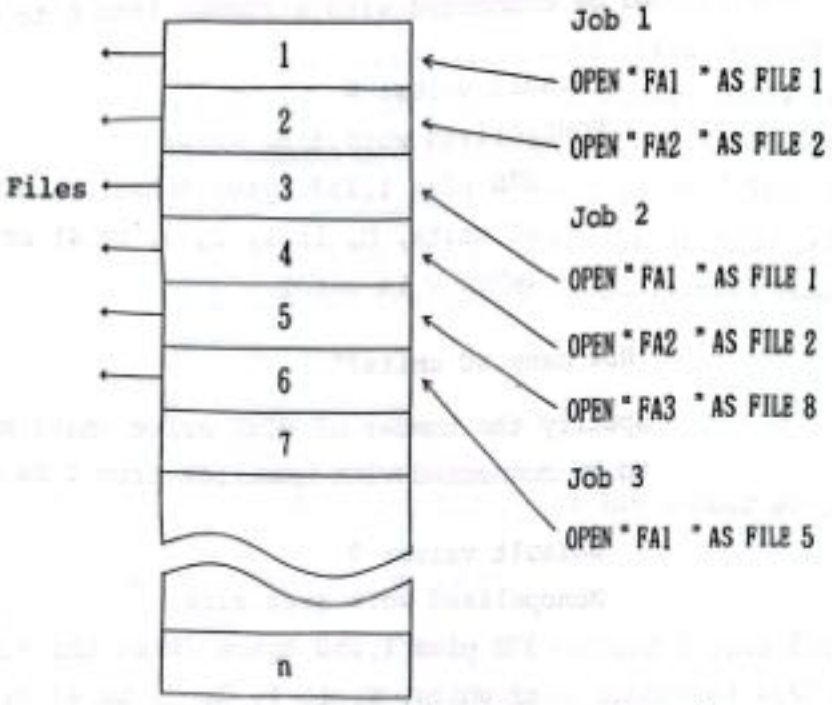
Note: When a number other than 0 is specified for WA or WB, the monopolized work area size is expressed as 37N plus 741 bytes.

5. "How many system channels?"

Specify the number of system channels.

Default value: 16
Monopolized work area size:
122N plus three bytes (N: Number of system channels)

The following figure shows the relationship between system channels and user channels:



System channels and user channels

Eight user channels from No. 1 to No. 8 are reserved for respective users (jobs). System channels are provided inside MDOS/XMS and any user channel receives data through a system channel. In the above figure, three user jobs use total six channels. These user channels are linked to free system channels in the order in which they were opened. When a user channel is closed, a system channel linked to the user channel is set free. If an attempt is made to open a user channel when there is no free system channel, an error message is displayed. So the number of system channels must be specified so as not to exceed the total number of user channels used by jobs at the same time.

6. "How many memory files?"

Specify the number of memory files.

Default value: 3
Monopolized work area size:
64N plus one bytes (N: Number of memory files)

The number of memory files does not indicate just that of memory files created by users (jobs). It also includes that of memory files created by the system, that is, it indicates the total number of memory files created by both the system and user jobs.

7. "User boot message?"

Specify a user bootstrap message.

Default value: blank
Monopolized work area size: 0

The user bootstrap message is displayed with the system bootstrap message when MDOS/XMS is bootstrapped. It must consist of up to 35 characters. If 36 or more characters are entered, the first 35 characters are only valid.

8. "Code conversion table file?"

In response to this message, the user can specify whether code conversion is to be made or not for each I/O device such as the main frame keyboard, a display unit, or a printer. If code conversion is required, enter the name of a code file prepared beforehand.
When the code file name is entered, the system inquires whether code conversion is to be made or not for all I/O devices specified in the SIO and SOUT files respectively when specifying as what devices SIO channels are to be used.
If no code file name is entered, the system recognizes that code conversion is not to be made and does not inquire that.
The medium for system generation contains the source file SORD.SR and object file SORD.COD as code file examples. For details on how to create a code file, see the manual entitled 'M243EX MDOS/XMS I/O Code Conversion Function for the M243EX Main Frame keyboard, display units, and printers'.

9. "Serial port (1 :SIN/SOUT 2:RT-20 3:VT-100
4 :KT-20)?"

Specify as what devices serial ports (A, B, C, and D) provided for the standard main frame are to be used respectively. The user must use any one of the following file or device names in specification:

. SIN/SOUT: Is to be specified when the serial port is to be used as a serial I/O port. The name to be used depends on the channel as follows:

Channel A: SINA, SOUTA
Channel B: SINB, SOUTB
Channel C: SINC, SOUTC
Channel D: SIND, SOUTD

. RT-20: Is to be specified when the serial port is to be connected to a RT-20 terminal.

. VT-100: Is to be specified when the serial port is to be connected to a VT-100 terminal. (VT-100 is a trademark of Digital Equipment Corporation of the U.S.A.)

. KT-20: Is to be specified when the serial port is to be connected to a KT-20 tablet.
Data entered from the KT-20 is processed in the same manner as that entered from a console keyboard. That is, data is processed on the 'first come, first served' basis regardless of from which device, the KT-20 or the console keyboard, the data is entered. The user can control the KT-20 by a program.
For details on the KT-20, see the kanji tablet KT-20 operation manual.

If any of the above names is not specified for a channel, that is, only the RETURN key is pressed, the channel is not supported by MDOS/XMS.
Default values for these channels are as follows:

Channel A	Unused
Channel B	Unused
Channel C	Unused
Channel D	Unused

The monopolized work area size depends on which name is specified as follows:

. When SIN/SOUT is specified

  204N plus 577 bytes (N: Number of channels for which the file name is specified)

  Example: When the file name is specified for Channels A, B, and C

  204 x 3 + 577 = 1,189 bytes

. When the RT-20 or VT-100 is specified

  621N plus 521 bytes (N: Number of terminals specified)

  The number of consoles is not included.

. When the KT-20 is specified

  0

[Input examples]

A=1, B=1, C=2, D=2

This assigns Channels A and B as serial I/O ports and Channels C and D as terminals.

A=2, D=4, C=1

This assigns Channel A as a terminal, Channel C as a serial I/O port, and Channel D as a tablet. Channel B is not supported by MDOS/XMS.

10. "SIO-n initialize data?" (n:A~H)

Specify initial data for serial ports A, B, C, and D provided for the standard CPU main frame. The user needs to enter data in response to input request messages displayed in the order from Channel A to D respectively. However, no input request message is displayed for a channel specified not to be used. So the user does not need to enter data for the channel. Each serial port is set to the state specified in response to this message when MDOS/XMS is bootstrapped.
The following six parameters must be specified as initial data:

. Baud rate (B)

Specify any one of 50, 75, 110, 150, 200, 300, 600, 1200, 2400, 4800, 9600, and 19200.

. Mode (M)

Specify either W (Wait) or I (Ignore).

. Parity

Specify N (No parity), O (Odd parity), or E (Even parity).

. Stop bit (S)

Specify 1, 1.5, or 2.

. Transmission data bit length (T)
Specify 5, 6, 7, or 8.

. Receipt data bit length (R)
Specify 5, 6, 7, or 8.

[Input example]

If an item is omitted, its default value shown in the following table is assumed.

	Baud rate	Mode	Parity	Stop bit	Transmission data bit length	Receipt data bit length
SIN/SOUT	9600	*	N	2	8	8
RT-20	9600	W	N	2	8	8
VT-100	4800	I	N	2	8	8
KT-20	2400	W	N	2	8	8

Fig. Default values of initial data for serial ports

*: The default value is W for CHannels A, C, and D, and it is I for Channel C.

When the RT-20, VT-100, or KT-20 is specified, the above default values should be specified respectively.

Monopolized work area size: 0

12. "SIO n:Code conversion YES or NO?"

This input request message is displayed only when the file name for I/O code conversion is specified in Item 8 and code conversion is specified for a printer in the file. That is, this message is for a channel specified as a serial I/O port in Item 7 and the POUT file. If code conversion is to be made, enter Y and if it is not to be made, enter N.

Default value: N
Monopolized work area size: 0

13. "Function table for terminal-n?" (n:0 ~ 4)

Specify a function table for each console and terminal. The contents of a file specified in response to this message is loaded into the function table for the console or terminal when the new MDOS/XMS is bootstrapped. This is the case with all console and terminals. The system displays input request messages in the order from Terminal 0 (console) to Terminal 4, so the user needs to enter a file name for each message. If the file extension .FNC is to be used, the user can omit it. Terminal No.1 is assigned to a terminal which was connected to the serial port A or its nearest port and then Terminal No.2, 3, and 4 are assigned in this order.
Any filename to be specified in response to this message must be those created by the FUNC command.

Default value: BASIC.FNC for both the console and terminals
Monopolized work area size: 0

14. "Auto start command line for terminal-0?"

This message requests the user to specify an automatic start command line and is for the main console and serial ports specified as terminals. By specifying the automatic start command line, the user can set the system to the READY state without a log-in operation and start one or more user jobs automatically. Following the above message, the system displays this message:

Auto start command line:

To specify only a log-in operation to set the system to the READY state, press the RETURN key. The message changes as follows:

Auto start command line: <LOGIN>

To set the system to the READY state and execute one or more commands automatically, enter a command line containing the commands.
To specify the command line beyond the limit of the maximum number of characters which can be entered, the user must create command files beforehand and enter the command file names (@command file name) at this point. The maximum number of characters which can be entered for the command line is 20. When the command line is terminated, the following message is repeated.

Auto start command line: Command line entered

If specification of the command line is not required, press the END key, or the CTRL and W keys at the same time.

[Notes]

. System generation must not be made while any other job is being executed.

. System generation requires two memory banks. Delete memory file created for system configuration when use of them is completed. Otherwise, an error may be caused.

. Do not make any forced interrupt such as hardware resetting while MDOS/XMS newly created by system generation is being written into the medium. The new MDOS/XMS does not operate normally.

. MDOS/XMS to be newly generated can accept system configuration different from the actual one. That is, a component which does not exist in the actual system can be specified and every system component must not be specified in the MDOS/XMS.
If this is the case, only specification of components which exist both in the actual system and the MDOS/XMS is made valid.

. The total numbers of bytes of the work area and of table buffers required by the above parameters slightly differs with MDOS/XMS versions.

. Do not connect the VT-20, VT-100, or KT-20 to the serial port B by the standard cable. If it is connected by the standard cable, the device does not operate normally because signals of Channel B are different from the other channels'.

8.2.7 MTBK and MTDL

The MTBK and MTDL utility program are used to transfer data from a winchester fixed disk drive unit to the magnetic tape unit MTU-80.

MTBK transfers all images from a disk at a time, whereas MTDL allows data to be transferred in units of a file. For details on MTBK and MTDL operations, see the following manual:

- Magnetic Tape Unit MTU-80 Operation Manual

8.2.8 MDUMP

The MDUMP command is used to transfer one or more files from a winchester fixed disk to a mini-floppy disk or standard floppy disk.

A winchester fixed has a large storage capacity. It is possible that a floppy disk does not have an enough space to store one file transferred from a winchester fixed disk. If an XFER command is used when this is the case, the error message 'DISK SPACE EXHAUSTED' is displayed and the file transfer fails.

But the MDUMP command allows the user to transfer a file requiring a large space to more than one floppy disk by separating the file.

Specifying a filename in a command line transfers the file only and specifying a directory name transfers all files registered in the directory.

MDUMP initializes the floppy disk to be used in the same manner as the MINIT command before file transfer. So the user should not use any floppy disk containing important files.

When initialization of the floppy disk is completed, a directory necessary for file transfer by MDUMP is created on the floppy disk. The directory name is MDMP01.DIR and the maximum directory size, depending on the floppy disk, is reserved.

A number of the last two digits of the directory name indicates a sequential number. If the file is transferred over two or more floppy disks, the directory of the second floppy disk is created under the name MDMP02.DIR.

When creation of the directory is completed, the names of files to be transferred are registered in the directory.

MDOS/XMS enables files to be transferred back from a floppy disk to a winchester fixed disk by the MLOAD command.

[Formats]

MDUMP disk drive unit number:[directory name]
filename floppy disk drive unit number:
MDUMP disk drive unit number:[directory name]
floppy disk drive unit number:

[Switches]

- Global switch --- None
- Local switch ---- None

[Examples]

The user can transfer one or more files either by specifying the filename or by specifying only the name of a directory in which the files are registered. In the latter case, all files registered in the directory are transferred.

1) The following MDUMP command specifies that the file DATA registered in the directory USER is to be transferred from the 20 MB winchester fixed disk drive unit No. 0 to the standard floppy disk drive unit No. 1.

```
MDUMP WB0:[USER]DATA FA1:
```

When this command is entered, the following message is displayed:

```
SET DUMP DISKETTE AND STRIKE "D"-KEY
```
(In response to this message, insert a floppy disk into the drive unit 1 and press the D key. When D is typed, the floppy disk is initialized and a directory is created.)

```
USER. DIR
 DATA
```
(MDUMP displays the specify directory name and searches for the specified file. When it is detected, MDUMP displays the filename and starts file transfer.)

```
DATA #DUMP DISKETTE DISK AREA FULL
SET DUMP DISKETTE AND STRIKE "D"-KEY
```
(If the floppy disk becomes full while the file DATA is being transferred, the system displays the above error message and requests the user to replace the floppy disk with another. Replace it and press the D key again. The new floppy disk is also initialized, a directory is created, and file transfer is continued.)

```
USER. DIR
 DATA
SET OS DISKETTE AND STRIKE "D"-KEY
```
(When the entire contents of the file DATA is transferred, the above message is displayed. In response to this message, remove the floppy disk, insert another floppy disk if required, and press the D key. The system returns to the READY prompt. If the entire contents of the file cannot be stored on the two floppy disks, the same error message as above is displayed. The user needs to continue the same operations as described above until transfer of the entire contents is completed.)

2) In this example, all files registered in the directory SUB are transferred from the 10 MB winchester fixed disk drive unit No. 0 to the mini-floppy disk drive unit No. 1.

```
MDUMP WA0:[SUB] FA1:
SET DUMP DISKETTE AND STRIKE "D"-KEY
SUB. DIR
 DATA1  (If the floppy disk also becomes full,
 DATA2
 DATA3
   |     the system displays the above error
   |     message again and requests the user to
   |     replace the second floppy disk with
   |     another.  Replace it and press the D
   |     key again.)
 DATA10 #DUMP DISKETTE FILE NAME FULL
SET DUMP DISKETTE AND STRIKE "D"-KEY
(When transfer of all files is completed, the
above message is displayed.  Remove the floppy
disk, insert a floppy disk if required, and
press the D key.  The system returns to the
READY prompt.

SUB. DIR
 DATA10
 DATA11
   |
   |
   |
 DATA15 #DUMP DISKETTE DISK AREA FULL
SET DUMP DISKETTE AND STRIKE "D"-KEY
(If the floppy disk also becomes full, the
system displays the above error message again
and requests the user to replace the second
floppy disk with another.  Replace it and press
the D key again.)
```

```
SUB. DIR
 DATA15
 DATA16
    |
    |
    |
    |
    |
SET OS DESKETTE AND STRIKE "D"-KEY
(When transfer of all files is completed, the
above message is displayed.  Remove the floppy
disk, insert a floppy disk if required, and
press the D key.  The system returns to the
READY prompt.)

READY
```

8.2.9 MLOAD

This command is used to return a file transferred to a floppy device to a Winchester fixed disk by an MDUMP command.

[Formats]

MLOAD floppy device number:filename disk device name:[directory name]
MLOAD floppy device number:disk device name:[directory name]

When a file name is not specified on a floppy device, only the file is transferred.

When a file name is not specified on a floppy device, all files registered in the transfer directory on a floppy device are transferred.

Since the floppy disk has already specified the transfer directory name, the directory name must not be specified.

Besides, when the directory name is not specified on the disk, the current default directory name is used.

[Examples]

1) When a file name "DATA is loaded on the directory "USER" of a 20 MB Winchester fixed disk from the standard floppy unit No. 1.

```
MLOAD FA1:DATA WB0:[USER]
SET DUMP DISKETTE AND STRIKE "D"-KEY
```

(Set the floppy device with the directory name "MDMP.DIR" on the floppy device to the specified drive and press the "D" key.

```
MDMP01.DIR
        DATA
```

(The directory name is then displayed. If the number of the directory name is not 01, an error message is displayed. The user should check the floppy disk before inserting it. If the correct floppy disk is inserted, the system searches for the specified file and transfers it.)

```
SET DUMP DISKETTE AND STRIKE "D"-KEY
```

(If the file DATA is stored over two or more floppy disks, the system displays the above message and requests the user to replace the first floppy disk with the second one. Replace it and press the D key.)

```
MDMP02.DIR
        DATA
SET OS DISKETTE AND STRIKE "D"-KEY
```

(When reading of the last floppy disk containing data of the file D is completed, the above message is displayed, remove the floppy disk, insert another disk if required, and press the D key. The system returns to the READY state.)

2) In this example, files are transferred from the mini-floppy disk drive unit No. 1 to the directory SUB on the 10 MB winchester fixed drive unit No. 0.

```
MLOAD FA1: WA0:[SUB]
SET DUMP DISKETTE AND STRIKE "D"-KEY
MDMP01.DIR
 DATA1
 DATA2
 DATA3 #FILE ALREADY: DELETE? (YES, NO)
```

(The system displays filenames registered in the floppy disk directory one after another and if the same filename already exists on the winchester fixed disk, it requests the user to specify whether the file is to be transferred or not by displaying the message shown on the line starting DATA3. If Y is typed, the file on the winchester fixed disk is overwritten by the floppy disk file. If N is typed, the floppy disk.)

```
 DATA9
SET DUMP DISKETTE AND STRIKE "D"-KEY
MDMP02.DIR
 DATA10
 DATA11
 DATA12
 DATA13
 DATA14
SET DUMP DISKETTE AND STRIKE "D"-KEY
MDMP03.dir
 DATA15
SET OS DESKETTE AND STRIKE "D"-KEY
READY
```

8.2.10 ADUMP

The ADUMP command is used to transfer all block images from a winchester fixed disk to mini-floppy or standard floppy disks.

Unlike the XFER and COPY commands, ADUMP cannot transfer data in units of a file and transfers the entire contents of a winchester fixed disk as written above. Usually, therefore, to transfer the entire contents of a 10 MB winchester fixed disk, 10 standard floppy disks are required and to transfer the entire contents of a 20 MB winchester fixed disk, 19 standard floppy disks are required. (Number of floppy disks required = total number of blocks of the winchester fixed disk ÷ number of user blocks excluding those for MDOS/XMS)

This command is used when saving the contents of a winchester fixed disk temporarily for creating a back-up disk and when copying the contents of a winchester fixed disk to another system.

When an ADUMP command is entered, the floppy disk is initialized first in the same manner as by the MINIT command. Therefore, the user should not use any floppy disk containing important files.

When initialization of the floppy disk is completed, the system then stores the first file in the contiguous file format. The contiguous file is created so as to use the floppy disk space to the full extent. As the filename, the winchester fixed disk drive unit name is used and a sequential number is added as the extension.

For example, if the following ADUMP command is entered, the filename 'WA. 01' is attached to the first floppy disk, 'WA. 02' is attached to the second disk, and this continues up to 'WA. 10'.

[Format]

ADUMP disk drive unit number:floppy disk drive unit number:

[Switches]

- Global switch --- None
- Local switch ---- None

[Examples]

In this example, all images of a 20 MB winchester fixed disk are transferred from the drive unit No. 0 to the standard floppy disk drive unit No. 1.

```
ADUMP WB0: FA1:
SET DUMP DISKETTE AND STRIKE "D"-KEY
```

(In response to this message, insert a first floppy disk into the specified drive unit 1 and press the D key.)

```
WB. 01
```

(The system initializes the floppy disk, and registers and displays the first filename, and starts file transfer.)

```
SET DUMP DISKETTE AND STRIKE "D"-KEY
```

(When the first floppy disk becomes full, the above message is displayed. Replace the first floppy disk with a second one and press the D key.)

WB. 02
(The system initializes the floppy disk, and registers and displays filenames, and resumes file transfer.)

(Repeat the above operations up to a tenth floppy disk.)

WB. 19
READY
(When transfer of all images of the winchester fixed disk is completed, the system returns to the READY state.)

If an error occurs on the winchester fixed disk during file transfer, an erroneous block number is displayed in the following format and data of the block is all replaced with NULL codes and transferred to the floppy disk:

SOURCE DISK: ERROR BLOCK XXXXX

If an error occurs on a floppy disk during file transfer, the system displays the following message and requests the user to replace the erroneous floppy disk by displaying the SET DUMP DISKETTE message:

DUMP DISKETTE: DISK FAIL

In response to the message, the user needs to replace the disk with another to continue file transfer.

Notes:

- If a floppy disk was initialized with the sector 1, data can be written into the floppy disk faster than any disk initialized with the default sector sequence 8. When writing one floppy disk of data, it takes three to four minutes for the latter disk, but it takes two and a half minutes for the former disk.
- If data in a system area block of a winchester fixed disk is destroyed, the data cannot be read by any command such as the LIST command. But the ADUMP program of Revision 00D or later can read and transfer such data.

8.2.11 ALOAD

The ALOAD command is used to transfer all images back to a winchester fixed disk from floppy disks. The images were transferred to the floppy disks by the ADUMP command.

[Format]

ALOAD floppy disk drive unit number: disk drive unit number:

[Switches]

- Global switch --- None
- Local switch ---- None

[Example]

In this example, data is transferred from the standard floppy disk drive unit No. 1 to the 20 MB winchester disk drive unit No. 0.

```
ALOAD FA1: WB0:
SET DUMP DISKETTE AND STRIKE "D"-KEY
(Insert the first floppy disk with the file WB.01
into the specified drive unit and press the D key.)

WB. 01
(When the D key is pressed, the system displays the
name of a floppy disk file to be transferred and
checks the filename registered on the floppy disk.
If the filename is correct, it starts data transfer.)

SET DUMP DISKETTE AND STRIKE "D"-KEY
(When ready of the floppy disk is completed, the
above message is displayed.  Replace the first
floppy disk with the second one and press the D
key.)
```

WB. 02

(The system checks the filename registered on the second floppy disk and if the filename is correct, it resumes data transfer.)

WB. 19

READY

(When transfer of all data is completed, the system returns to the READY state.)

If an attempt is made to transfer data from floppy disks in a wrong order, the system displays the following message and then the SET DUMP DISKETTE message again.

SOURCE DISKETTE : ERROR BLOCK XXXXX

Replace the wrong floppy disk with the correct one and press the D key.

If an error occurs on a floppy disk during data transfer, the system displays an erroneous block number in the following format and replaces all data on the block with NULL codes:

DESTINATION DISKETTE : ERROR BLOCK XXXXX

If an error occurs on the winchester fixed disk during data transfer, the following message is displayed and no data is written into the erroneous block:

DESTINATION DISK : ERROR BLOCK XXXXX

8.2.12 ACOMP

The ACOMP command is used to compare data transferred to floppy disks by the ADUMP command with the original data on a winchester fixed disk.

[Format]

ACOMP floppy disk drive unit number:disk drive unit number:

[Switches]

- Global switch --- None
- Local switch ---- None

[Example]

In this example, floppy disk data in the standard floppy disk drive unit No. 1 is compared with the original data on a 20 MB winchester fixed disk in Unit 0.

```
ACOM FA1: WB0:
```
(Insert the first floppy disk into Unit 1 and press the D key.)

```
WB. 01
```
(The system displays the name of a floppy disk file to be compared and checks the filename registered on the floppy disk. If the filename is correct, it starts data comparison.)

```
SET DUMP DISKETTE AND STRIKE "D"-KEY
```
(When comparison of the first floppy disk is completed, the above message is displayed. Replace the first floppy disk with the second one and press the D key.)

WB. 02

(The system checks the filename registered on the disk and if it is correct, it resumes data comparison.)

WB. 19

READY

(When comparison of all data is completed, the system returns to the READY state.)

If floppy disks are inserted in a wrong order, the system displays the following message and then returns to the SET DUMP DISKETTE message:

SOURCE DISKETTE : FILE DOES NOT EXIST

Replace the wrong floppy disk with the correct one and press the D key.

If inconsistent data is detected during data comparison, the number of a block in which the inconsistent data exists is displayed in the following format:

ERROR BLOCK XXXXX

Data is displayed in units of a byte, so the error message is also displayed each time an erroneous byte is detected regardless of whether they are in same block.

APPENDIX A MAIN FRAME DIP SWITCHES

A 8-bit DIP switch is provided at the back of the M243EX main frame. The DIP switch is used to set the system to a required state. The system recognizes the switch settings and processes data according to the settings.

The DIP switch is set when shipping the system from a SORD factory and its settings differ with the system. This appendix describes switches which can be handled by the user.

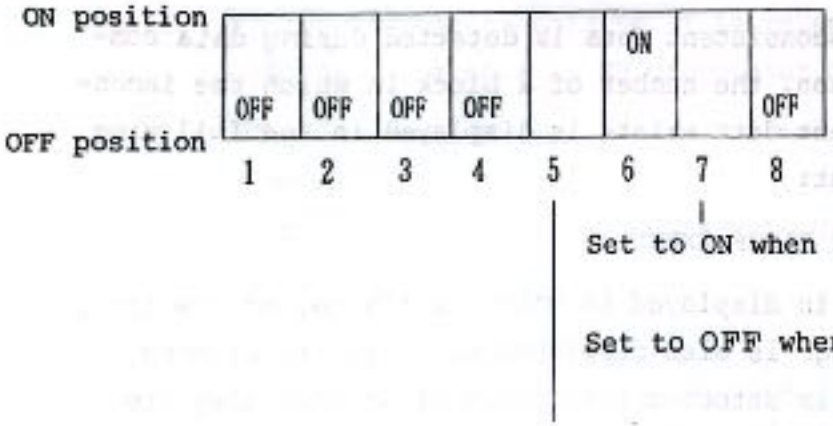


Fig. Settings of DIP switches
(Switches other than 5 and 7 are fixed.)

Setting the DIP switch 7 for display units

The DIP switch is set according to the type of a display unit to be connected to the system. If it is not set correctly, processing shown in Table A-2 is performed. This is because MDOS/XMS performs processing for the display unit according to the switch setting. If the DIP switch 7 is set to the OFF position although a color display unit is connected, MDOS/XMS performs processing for a monochrome display unit. As a result, data is displayed in black and white on the color display unit.

Setting of the DIP switch 7 / Display unit connected	ON	OFF
Color	Processing for color display units is performed.	MDOS/XMS recognizes that a monochrome display unit is connected. As a result, characters are displayed in white and graphic images are displayed in red.
Monochrome	MDOS/XMS recognizes that a color display unit is connected and performs processing for it. As a result, data is displayed in eight different brightness levels corresponding according to specified colors.	Processing for monochrome display units is performed.

Fig. Relationship between switch settings and display units

The eight brightness levels for monochrome display units are corresponded to eight colors used for color display units. The function for displaying data in different brightness levels makes monochrome display units each to watch and is useful when debugging programs on the color display unit.

Eight brightness levels correspond to colors as follows:

Color	Brightness (higher, brighter)
White	7
Yellow	6
Magenta	5
Red	4
Cyanic	3
Green	2
Blue	1
Black	0

Character display attributes are handled as follows:

Display attribute \ Display unit	Color	Monochrome
Highlighting	Not made.	Made.
Underline	Displayed in white.	Displayed.
Blinking	Made.	Made.
Reverse display	Made.	Made.
Color specification	Made.	When the DIP switch 7 is set to the OFF position, it is not made. When the switch is set to the ON position, it is made and data is displayed in eight different brightness levels.

Setting the DIP switch 5

The DIP switch 5 is used to specify from which device, a floppy disk or winchester fixed disk, MDOS/XMS is to be bootstrapped.

When the DIP switch is set to the ON position, MDOS/XMS is bootstrapped from the floppy disk regardless of whether a winchester fixed disk drive unit is connected. To bootstrap MDOS/XMS from the winchester fixed disk, the DIP switch must be set to the OFF position beforehand.

Figure shows the system device selection flow in bootstrapping MDOS/XMS.

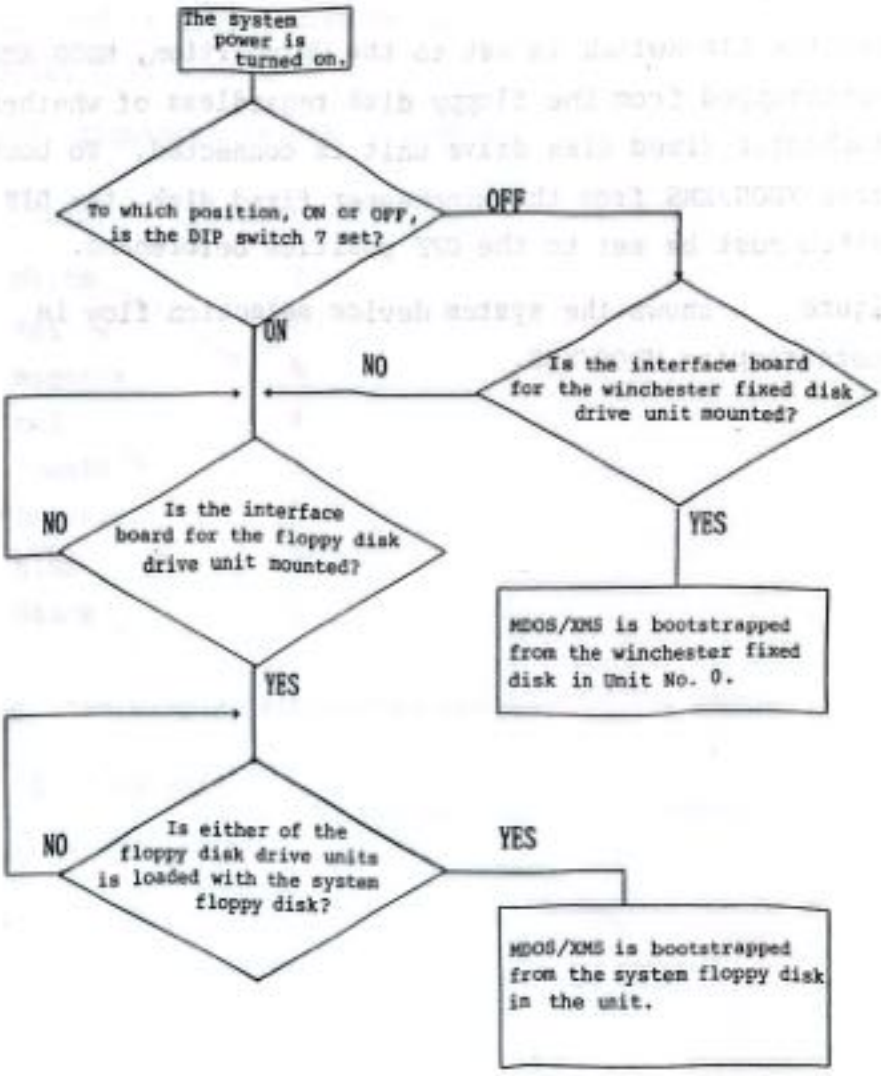

The system power is turned on.
To which position, ON or OFF, is the DIP switch 7 set?
OFF
ON
Is the interface board for the winchester fixed disk drive unit mounted?
NO
YES
MDOS/XMS is bootstrapped from the winchester fixed disk in Unit No. 0.
Is the interface board for the floppy disk drive unit mounted?
NO
YES
Is either of the floppy disk drive units is loaded with the system floppy disk?
NO
YES
MDOS/XMS is bootstrapped from the system floppy disk in the unit.

Fig. System device selection flow

- END -